Русифицированная версия 1.22.051.

Mission Director – Руководство для начинающего разработчика

Вступление

В предыдущих играх серии X Series, миссии (как сам сюжет, так и различные дополнительно-вспомогательные) требовали большого искусства для их написания, их было довольно сложно тестировать, что способствовало появлению ошибок, и они были ограничены тем, что писать их приходилось с использованием KC. В результате мы получали редко устраняемые баги в сюжете, и очень малое количество внесюжетных миссий.

Главная задача Mission Director-а - создание простого и удобного интерфейса для разработки и написания миссий, который сможет использовать и не программист. Второй (но не менее важной) задачей является возможность обновления и дополнения миссий без выпуска новых патчей к игре.

В качестве основы для разработки миссий была выбрана платформа **XML**, легко обеспечивающая функциональность работы с "plug-in"-ми, и более доступная для не программистов. Добро пожаловать …

Используя эту базу, вы легко сможете формировать события в игре, которые в комплексе сложатся в миссию, которую необходимо будет выполнить игроку, тесно взаимодействуя с игровыми объектами, такими как станции и корабли.

Mission Director прост для освоения, от вас потребуется только элементарное понимание "data logic" для написания миссий. В процессе использования Mission Director ваше мастерство владения этим инструментом будет расти, и соответственно будет усложняться "код", следовательно будет расти сложность и интересность миссий, которые вы напишете.

Миссии имеют жесткую и четкую структуру, что приводит к простоте их отладки, тестирования и исправления, без необходимости модифицировать игровой код. Вы легко сможете создать простое событие, увидеть и проверить его в игре, исправить, и тут же увидеть изменения, просто рестартовав Mission Director, или перезагрузив игру. Эта беспрецедентная гибкость - одна из основных особенностей Mission Director, приводящая к ускорению в разработке миссий, прямо на глазах.

Примеры

Некоторое количество миссий было создано в процессе разработки самого движка MD. Использование этих миссий для наглядного изучения поведения движка MD, их модификация, и изучение результатов такой модификации - наилучший путь к пониманию сущности MD. Эти миссии не только помогут вам разобраться в том, как все работает, но и предоставят в ваше распоряжение библиотеку примеров, наш передовой опыт, руководство по достижению необходимого вам результата.

Примеры окажут вам помощь и при написании своих миссий, подобных тем, что в них описаны, и вы можете изменять их по своему усмотрению, редактировать и использовать отдельные кусочки их кода для написания своих собственных миссий. Это не будет являться плагиатом с вашей стороны, мы воспримем использование наших примеров как комплимент. Данное руководство призвано дать вам описание ключевых элементов миссий, и затем, используя обучающие примеры, пошагово обучить вас писать миссии. после того, как вы разберетесь со всеми примерами, вы намного лучше станете понимать принцип работы MD, и будете иметь достаточно знаний и опыта для самостоятельного написания миссий.

Ограничения

Некоторый опыт в программировании поможет вам лучше и быстрее разобраться с функциональностью MD, однако он не является обязательным для понимания. И даже, в некоторых случаях может помешать, так как MD использует событийно-ориентированный подход, в отличии от привычного для программистов объектно-ориентированного подхода. Это руководство ориентировано на тех, кто не обладает, или обладает начальными знаниями в программировании, оно написано для новичков. Однако, подразумевается что вы прекрасно знаете Вселенную X и обладаете большим опытом жизни в ней. И этого абсолютно достаточно для любого, кто хочет приступить к использованию MD.

Disclaimer

В настоящий момент, данная версия Mission Developer является тестовой и не поддерживается фирмой Egosoft. Помощь может быть получена от добровольцев на форуме, но проблемы в коде или схемах не будут исправлены. Есть некоторое количество мелких недоработок, однако они не должны повлиять на выполнение файлов Миссий при использовании Mission Director-a.

Установка

Установка MD

Mission Director активируется вместе с редактором скриптов (Script Editor), путем изменения имени игрока на "Thereshallbewings". Так же вам необходимо убедиться, что в папке с игрой у вас присутствует папка с именем „director". Для того, чтобы использовать миссии в игре, вам достаточно того, чтобы файлы с их описаниями находились в указанной папке. Однако, если вы хотите редактировать и писать миссии самостоятельно, вам необходимо убедиться в наличии в указанной папке некоторого количества вспомогательных файлов, список которых приведен ниже.

Расположение файлов MD

Файлы Mission Director-a, которые необходимы для редактирования и написания миссий приведены на Рисунке 1:

Шесть файлов, расположенных в верхней части Рисунка 1 обязательно должны находиться в папке director, если вы желаете самостоятельно редактировать и создавать миссии. Произвольное количество XML-файлов с описанием миссий может находиться в данной папке, и игра будет пытаться загрузить их все, в тот момент, когда вы загружаете или стартуете игру. Для того, чтобы избежать проблем и конфликтов, обязательно

убедитесь в том, что все Эпизоды(cues) в этих файлах имеют уникальные идентификаторы. Два других файла, представленных на рисунке ниже - это примеры таки XML-файлов.

Краткое описание функционала некоторых файлов из папки *directorr*:

director.htm

Это очень нужный файл, особенно новичку. Открыв его в Internet Explorer-е, вы получите доступ к списку всех "команд" MD, а так же с писку их параметров и атрибутов. Так же вы получите список большого количества доступных констант и переменных. Используя фильтрацию вы сможете ограничить этот список только необходимыми "командами", константами и переменными. Если Internet Explorer выдает сообщение о блокировке активного содержимого, вам необходимо разрешить использование элементов ActiveX, для полноценного использования данного файла.

director.xsl & dirschema.xsl

Эти файлы содержат правила форматирования для всех ключевых слов MD, как для XML, так и для HTML файлов в данной папке.

director.xsd & dirobjdb.xsd

Эти два файла описывают правила именования, использования ключевых слов при редактировании миссий во внешнем редакторе XML, и фактически содержат все доступные ключевые слова, команды, типы объектов и константы. Файл dirobjdb.xsd предоставляет вам доступ ко всем константам описывающим все игровые объекты в X3.

Инструменты для редактирования XML-файлов.

Существует большое количество свободных и коммерческих XML-редакторов, и они доступны в Интернете. Однако данное руководство ориентировано на использование Visual Studio 2005 Web Developer Express edition, который полностью поддерживает используемые в MD XML-схемы, хотя и не содержит окна древовидного просмотра. Он доступен для свободного скачивания..

XML-редактор миссий MD должен полностью поддерживать XML-схемы (не DTDs) и стили, так же хорошо, как и обеспечивать базовые основы редактирования XML, а так же желательна поддержка расцветки синтаксиса, авто-дополнения, древовидного просмотра и многого другого. Для обеспечения всех разработчиков оптимальной средой разработки миссий, мы им настоятельно рекомендуем использовать Visual Studio 2005 Web Developer Express. Если все разработчики используют одинаковые инструменты, то решение их проблем со стороны поддержки упрощается, и они больше времени смогут уделить непосредственно созданию миссий.

Создание нового файла миссий Mission Director -а

Сейчас мы попытаемся описать процесс создания нового файла миссий MD.

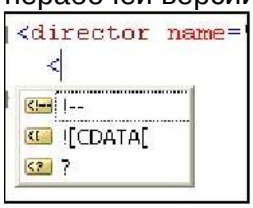
Откройте файл *Emptytemplate.xml*, расположенный в подпапке *samples* вашей папки *director*. Вы можете использовать этот файл для создания любой миссии. Однако, не забудьте сразу сохранить его пд новым именем, и убедитьс я, что сохраненный вами файл находится **в папке *director***.

```
<?xml version="1.0" encoding="iso-8859-1" ?>
<?xml-stylesheet href="director.xsl" type="text/xsl" ?>
<director name="test" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="director.xsd">
```

Проблемы при создании файла миссий

Если вы пытаетесь редактировать файл миссий, который расположен в папке отличной от папки director , то вы обнаружите, что авто -подстановка ключевых слов, согласно схеме MD, не работает, и будете наблюдать картину, аналогичную Рисунку 3. Есть два пути решения этой проблемы. Первый - вы должны скопировать файл миссий в папку director, и второй - вы должны скопировать все схемы и связанные с ними файлы в папку с файлом миссий.

Практически обязательным является создание резервных копий файлов миссий, желательно в другой папке. Это нужно для того, чтобы вы всегда могли при получении нерабочей версии миссии вернуться к ранее сохраненной рабочей версии.



Если, как на Рисунке 4, файл *director.xsd* выделен как ошибочный, то это значит, что в папке с файлом миссий опять же отсутствуют необходимые файлы, и вам надо либо скопировать файл миссий в папку *director*, либо скопировать все служебные файлы из папки *director*, однако, вам тогда придется следить за актуальностью этих фай лов в вашей рабочей папке.

Правила форматирования

Не зависимо от функциональности используемого вами XML -редактора, есть несколько основных правил форматирования файлов миссий, обеспечивающих корректную их загрузку и работу.

Обязательно используйте отступы для корректного отображения иерархии используемых XML -тегов и функций.

```
<parent_node>
   <first_subnode>
      <second_subnode>
         <nested_function/>
      </second_subnode>
   </first_subnode>
</parent_node>
```

Важно помнить о том, что все открытые XML -теги обязательно должны быть закрыты в последствии. Закрытие тегов может быть осуществлено двумя путями.

Первый - используется в случае если тег не имеет дочерних элементов, тогда он закрывается в конце текущей строки, например:
`<find_sector x="0" y="0" name="kingdomend"/>`.

Второй - используется в тех случаях, когда тег содержит зависимые дочерние элементы, как отображено в приведенном чуть выше примере, теги `<parent_node>`, `<first_subnode>` и `<second_subnode>` имеют зависимые дочерние элементы, и они закрываются отдельной строкой, повторяющей закрываемы тег, но с символом "/" перед именем тега, например:

`</parent_node>`.

Эпизоды <cues>

Тег эпизодов(cues) определяет раздел описания основных элементов в структуре миссий - эпизодов. Он может в себе содержать только теги вида `<cue>`, т.е. эпизод. Внутри тега `<cues>` может содержаться множество тегов <cue>, каждый из которых может содержать в себе так же вложенный тег <cues>.

```
<?xml version="1.0"?>
<?xml-stylesheet href="director
<director name="test" xmlns:xsi
   <cues></cues>
</director>
```

Если вы установите курсор между `<cues>` и `</cues>` (см. Рис.5) и нажмете Энтер, то курсор перейдет в новую пустую строку и в станет в позицию начала следующего вложенного тега, т.е. эпизода `<cue>`.

Эпизод <cue>

Тег <cue> - эпизод является основным элементом описания миссий, содержит в себе описание событий и действия, которые происходят, должны происходить в игре. Он имеет достаточно объемный список параметров, как показано на рисунке 6.

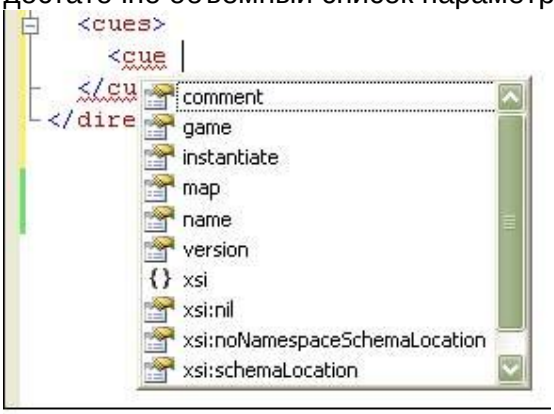


Figure 6 <cue>

Атрибут "instantiate" достаточно сложен для понимания и потому мы при водим его описание в Приложении 1.

Каждый эпизод-**cue** имеет четыре секции. Вот они:

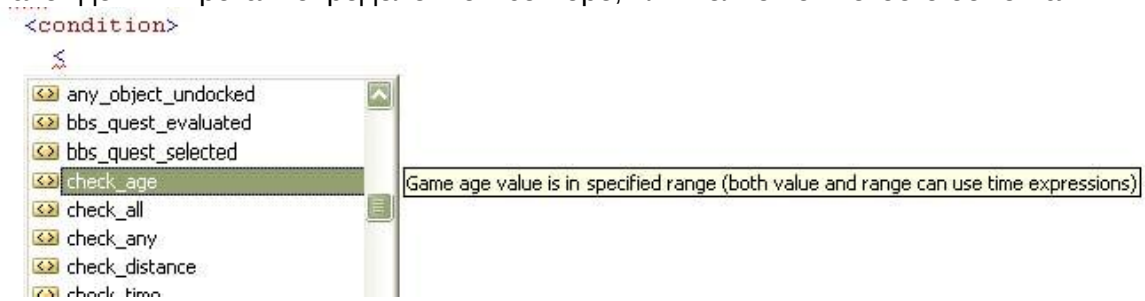
Секция (Sub-node)	Наличие обязательно	Допустимо наличие нескольких подразделов данного типа	Примечание
<condition> - условия активации	нет	нет	Если данная секция отсутствует, то эпизод выполняется автоматически и безусловно.
<timing> - хронометраж	нет	нет	Если данная секция отсутствует, то эпизод выполняется мгновенно, при старте.
<action> - действия	нет	нет	Если данная секция отсутствует, то эпизод может перейти в статус активного (в зависимости от предыдущих подразделов), но никакие действия выполнены не будут
<cues> - вложенные эпизоды	нет	нет	Вложенные эпизоды для данного эпизода.

Таблица 1. Подразделы тега(раздела) эпизод(<que>).

В Таблице 1 приведены все четыре секции, которые составляют из себя эпизод как таковой. И именно с ними вам предстоит работать при написании миссий. Секция действий <action> выполняется в указанное в секции <timing> время, при соблюдении условий, приведенных в секции <condition>. Вложенные эпизоды <cues> могут быть активированы в результате выполнения действия из секции <action>, с учетом собственных условий, времени выполнения и действий.

Условия активации <condition>

Секция условий активации (<condition>) может содержать одно или несколько условий, которые должны быть соблюдены для активации эпизода. Эти условия могут быть основаны на проверке простых значений переменной или об ъекта, принадлежности их к некоему диапазону значений, как например, время прошедшее с начала игры или сумма денег на основном счету игрока. Так же условия могут базироваться на более комплексной информации об игроке, например, наличии у игрока определенно го корабля или кораблей, нахождении игрока в определенном секторе, или вблизи от некоего объекта.



Другим, довольно таки важным классом используемых условий, являются условия, базируемые на событиях, происход ящих в игре, как например, события возникающие в момент нацеливания, атаки или уничтожения игрового объекта, либо события связанные с приемом либо отклонением игроком определенных ББС квестов. Условия могут группироваться по признакам либо обязательного со блюдения всех условий в группе, либо только хотя бы одного из них. При этом данная группа условий сама может входить в более высокоуровневую группу условий.

Таким образом могут быть использованы два основных способа группировки условий - по обязательному соблюдению всех (используется тег <check_all>) или хотя бы одного (используется тег <check_any>). **Важно помнить, что при использовании тега <check_all>, проверка условий на соблюдение идет последовательно. И если какое - либо условие не соблюдается, следую щие за ним - не проверяются. Поэтому, если вы в одной группе проверяете условия на появление события и на значения, обязательно размещайте условия на появление события в начале группы, чтобы избежать частой проверки значений.**

Так же условием активации эп изода может быть статус другого эпизода. К примеру, текущий эпизод может быть активирован только по успешному завершении "родительского"("parent") эпизода (при использовании условия <cue_is_complete cue="parent"/>). В то время как для главного эпизода усл овием активации может стать довольно простое условие, как например - время с начала текущей игры (см. рисунок 7).

В приведенном ниже примере происходит проверка заданного условия и при его выполнении - активации эпизода, то есть выполнение секции действий .

В данном случае условие будет соблюдено (и эпизод активирован) в промежуток времени между 5-й и 7-й секундой с начала игры.

```
<condition>
  <check_age value="{player.age}" min="5s" max="7s"/>
</condition>
```

Когда используются константы - единицы времени, аббревиатуры s, m и h используются для секунд, минут и часов.

Переменные (variables)

В приведенном ниже примере выражение `{player.age}` использовано как значение атрибута `value=""`. Данное выражение является переменной, и мы постараемся сейчас дать вам немного информации о переменных в MD и правилах их использования.

```
<condition>
  <check_age value="{player.age}" min="5s" max="7s" />
</condition>
```

Существует множество различных переменных, из идентификаторо в которых сразу становится ясно, какое значение они содержат, и которые напрямую связаны с игровыми объектами и значениями их параметров. Например: `{player.age}`, `{player.ship}` или `{player.money}`. Переменная `{player.sector.race.name}` может быть использована для получения информации о том, кто владеет сектором, в котором в данный момент находится игрок.

Так же существует другой класс переменных, они подобны описанным ранее, однако содержат отсылку к определенному пользователем элементу, который отображается как субъект переменной, идентификатор такого субъекта отделен от идентификатора переменной с помощью символа "`@`". Например: `{object.pilot@object}` – если вы создали корабль и желаете получить доступ к имени пилота этого корабля, вам необходимо указать идентификатор объекта после символа "`@`" , т.е. `{object.pilot@myship}` . Использовать данную переменную можно к примеру при исполнении следующего действия

```
<incoming_message author="{object.pilot@myship}" text="test"/>.
```

Важно учитывать, при создании и использов ании переменных Mission Director-а, что существует два вида переменных - "глобальные" и "локальные". Если переменная создается как глобальная, это означает, что она(и ее значение) будут доступны во всех Миссиях и Эпизодах. Если же переменная создается как глобальная, это означает, существует и ее значение доступно только во время существования Эпизода, с которым она была связана, и доступна только для внутри текущей миссии.

```
<cue name="createopponent"> <-The cue we're creating a ship in
   ...
   <create_ship name="opponent" <-This creates the object as a global
   <create_ship name="this.opponent" <-This creates it as a local variable
```

Префикс `"this."`, из приведенного выше примера, может быть использован только для привязывания создаваемого объекта и доступа к переменной внутри текущего Эпизода, т.е. для создания локальной переменной. Если вам необходимо использовать значение такой переменной во вложенных Эпизодах, доступ к ней осуществляется через указание имени Эпизода, в котором она была создана и привязана. В случае использования переменной, созданной в приведенном выше примере, доступ к ней должен быть осуществлен следующим образом…

```
<incoming_message author={object.pilot@createopponent.opponent} text="{1081,16}"/>
```

Т.е. мы использовали имя Эпизода `"createopponent"`, совместно с идентификатором переменной.

Дополнительная информация об использовании переменных:

Обычно для доступа к экземплярам, индексам и значениям переменных внутри Эпизодов используется имя Эпизода. Для сюжетных и других миссий, кото рые существуют в единственном экземпляре, этого вполне достаточно. Однако для миссий (Эпизодов), для которых допускается существование нескольких экземпляров в единицу времени (к примеру - миссии ББС) требуется способ более четкого определения, к какому и з экземпляров мы хотим обратиться. При попытке идентификации конкретного экземпляра, Mission Director пытается использовать текущий Эпизод, потом его владельца, и так до самого верха иерархической структуры Эпизодов. При этом, при наличии ответвлений, осущ ествляется попытка перейти к поиску сверху вниз, в каждом найденном ответвлении.

Если в процессе поиска не было найдено ни одного экземпляра, то MD осуществит поиск вниз по иерархии во всей иерархической структуре Эпизодов. Хотя поиск может окончится неудачей (в тоже время, для экземпляров Эпизодов исходный Эпизод всегда определен). Для того, чтобы избежать описанных выше сложностей с поиском экземпляра Эпизода, вам достаточно использовать префикс `this.` для указания текущего Эпизода, и `parent.` - для Эпизода владельца.

Хронометраж <timing>

Секция хронометража служит для определения, через какой промежуток времени после соблюдения Условий Активации Эпизод будет активирован и будет выполнено содержимое секции Действий. Так же в данной секции можно определить периодичность активации Эпизода. При этом в качестве параметров могут быть использованы как фиксированные, так и случайные значения переменных, лежащие в заданных интервалах значений, которые могут быть как фиксированными, так и сл учайными. В эпизоде может быть только одна секция Хронометража, и она не является обязательной, если действия должны быть произведены сразу после выполнения условий, и всего один раз.

Секция (Sub-node)	Описание
<count>	Количество повторений выполнения секции Действий.
<time>	Период времени, по истечении которого должна быть выполнена секция Действий, при соблюдении Условий активации.
<interval>	Интервал между исполнениями секции Действий
<params>	Использование в значений параметров из библиотечного эпизода.

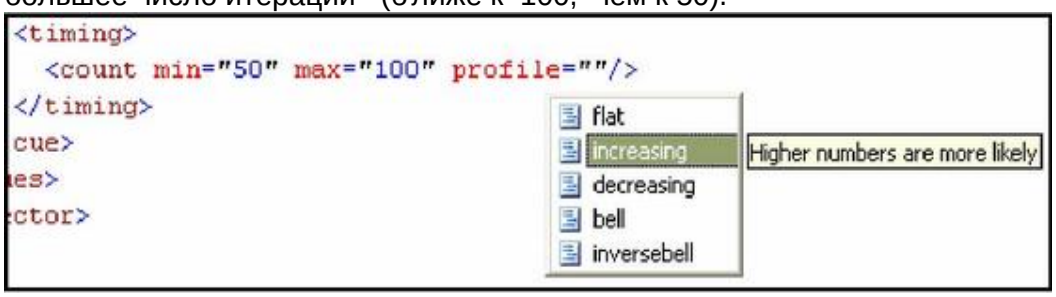
Таблица 2

В примере на рисунке 8 секция Действий будет выполнена через 5 -10 секунд после соблюдения Условий Активации Эпизода.

```
<timing>
  <time min="5s" max="10s"/>
</timing>
```

Рисунок 8. Пример секции Хронометража (<timing>)

В примере на Рисунке 9 секций действий будет выполнена 50 -100 раз, причем конкретное значение будет выбрано согласно одному и профайлов рандомизации, в данном случае использован профайл „increasing " , означающий, что скорее всего будет выбрано большее число итераций (б лиже к 100, чем к 50).



В примере на рисунке 10 выполнение секции Действий будет повторяться с интервалом в 5 секунд.

```
<timing>
  <time min="5s" max="60s"/>
  <interval min="5s"/>
</timing>
```

Рисунок 10. Использование подсекции interval.

Единицы измерения(Units of time and distance)

Стандартные единицы измерения времени и расстояния, и их соответствие внутренним единицам измерения MD.

distancem	100m	Переводит метры во внутренние единицы расстояния
distancekm	{value@this.xpos}km	Переводит километры во внутренние единицы расстояния
timed	5d	Переводит дни в миллисекунды
timeh	12h	Переводит часы в миллисекунды
timem	{value@this.delay}m	Переводит минуты в миллисекунды
times	({value@this.delay}*10)s	Переводит секунды в миллисекунды
m:s.ms	1:30.500	Переводит минуты/секунды/миллисекунды в миллисекунды
h:m:s.ms	1:30:0.0	Переводит часы/минуты/секунды/миллисекунды в миллисекунды

При записи значений времени, следует помнит, значения часов, минут, секунд и миллисекунд не должны содержать ведущие нули, т.е. это фактические значения, а не дробная часть числа. Учтите, что при такой записи невозможно задавать значение дней. Максимально значение времени, которое можно задать и получить таким способом 24 дня. Однако переменная `{player.age}` может принимать значения превышающие 24 дня, однако с точностью только до секунд.

Выражения (expressions)

Из переменных и констант можно составлять выражения, применяя простейшие математические действия. К примеру, вам надо проверить, не превышает ли указанная сумма денег сумму на счету игрока, как минимум на 1000 Кр:

```
<check_value value="{this.cashsum}" min="{player.money}+1000"/>
```

Ниже мы приводим простейшие математические операторы, которые могут быть применены к переменным и константам, имеющим числовой формат:

Оператор	Пример	Описание
x+y	{value@this.mylocalvalue1}+{value@this.mylocalvalue2}+2	Сложение
x-y	{value@this.mylocalvalue1}-{value@this.mylocalvalue2}	Вычитание
x*y	{counter@myloop}*10	Умножение
x/y	100/{value@this.mylocalvalue}	Деление (деление на 0 даст в результате 0)

(x)	(100+{value@this.mylocalvalue})/10	Выражения в скобках вычисляются первыми

Операторы в порядке понижения приоритетов исполнения: ( ), затем * и / и только затем + и -. Операторы одного приоритета исполняются в порядке слева - направо.

Действия <action>

Секция Действий эпизода содержит все необходимые действия, которые должны быть выполнены при соблюдении Условий Активации и согласно Хронометражу. Так же не стоит забывать, что секция Действий может быть выполнена более одного раза, в количестве итераций и с интервалом между ними, заданными в секции Хронометража. Эпизод может содержать одно действие, как например добавление некоей суммы на счет игрока, отправку сообщения игроку, создание одного или целой группы кораблей, а может содержать и множество различных действий. Все действия могут бы ть выполнены последовательно, друг за другом, но вы можете задать условия выполнения только одного из них, выбранном произвольно.

Как и в случае со списком возможных Условий Активации, при редактировании се кции Действий, вам доступен список возможных простых и сложных действий, и инструменты для их объединения между собой. Однако всегда следует помнить, что список Действий - это **не** программа. Обычные для программирования структуры, как например циклы и услов ные операторы здесь не существуют, в виде, привычном для программиста. Конечно, секции Хронометража и Условий Активации могут быть использованы для создания циклов и условных операторов, однако существуют и другие способы добиться нужного результата, без такого усложнения структуры Эпизодов. Многие действия могут быть выполнены по мере соблюдения новых условий, однако при этом могут располагаться в одном месте, описаны один раз, с минимальными повторениями и нагрузками на систему.

В каждом эпизоде может со держаться только одна секция Действий, поэтому, если требуется выполнить целый набор действий, то эти действия должны быть сгруппированы либо с помощью тега <do_all>, означающего, что будут выполнены все действия в группе, либо с помощью тега <do_any>, озн ачающего, что будет выполнено только одно действие из группы. Если же секция Действий не содержит ни того, ни другого тега - то она может содержать только одно возможное действие. Кроме того, каждый из тегов (<do_all> и <do_any>) может содержаться в друго м, что дает широкие возможности в описании возможных действий. Т.е. их использование аналогично использованию тегов <check_all> и <check_any> в секции Условий Активации. Далее, вы более подробно расскажем об использовании этих тегов.

Так же вы можете использовать довольно интересный тег `<do_if>`, который позволяет выполнить заданные действия, только при соблюдении заданного условия, когда одно значение эквивалентно другому. Так же вы можете использовать более развитый аналог данного тега, а именно - `<do_choose>`, который позволит вам провести проверку на соответствие нескольким значениям, а также содержит секцию "иначе"(otherwise), которая будет выполнена, если не одно из заданных вами условий не будет соблюдено. Вложенными секциями тега `<do_choose>` являются `<do_when>` и `<do_otherwise>`.

В приведенном ниже примере, используя тег `<do_any>`, мы создаем пустое сообщение для игрока. Когда Условия Активации эпизода будут соблюдены, одно из трех, произвольно выбранных, сообщений будет отправлено игроку. Обычно в кач естве параметра для тега-действия `<incoming_message>`, описывающего автора сообщения, используется строка текста, либо - идентификатор текста из текстового файла. Аналогично может быть задан и сам текст сообщения.

```
<action>
  <do_any>
    <incoming_message author="Shipboard Computer" text="Hello World"/>
    <incoming_message author="{1323,119}" text="Welcome to MD"/>
    <incoming_message author="{1323,119" text="{1278,305)"/>
  </do_any>
</action>
```

После описания секции Действий данного эпизода, вы можете описать вложенные эпизоды(`<cues>`), либо завершить описание данного эпизода(`</cue>`).

Пример № 1. Здравствуй, Мир! (Hello, World!)

Пример простой Миссии из одного Эпизода:

```
<?xml version="1.0"?>
<?xml-stylesheet href="director.xsl" type="text/xsl" ?>
<director name="test" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="director.xsd">
   <cues>
      <cue name="evaluator" game="all" version="1" comment="this is only so we know
      what's happening in this cue">
         <condition>
            <check_age value="{player.age}" min="5s" max="7s" />
         </condition>
         <timing>
            <time min="5s" max="10s"/>
         </timing>
         <action>
            <do_any>
               <incoming_message author="Shipboard Computer" text="Hello World"/>
               <incoming_message author="{1323,119}" text="Welcome to MD"/>
               <incoming_message author="{1323,119}" text="{1278,305)"/>
            </do_any>
         </action>
      </cue>
   </cues>
</director>
```

Тестирование Вашего первого файла миссий

После того как Вы завершили написание своей первой миссии, убедитесь, что файл Вами сохранен, и сохранен в подпапку director в папке с игрой. Так же Вам следует убедиться, что ни один из написанных Вами Эпизодов не имеет имени, аналогичного уже имеющимся в XML-файлах миссий в подпапке director.

Что ж, теперь следует запустить игру. Стартуйте новую игру, и смените имя игрока на *Thereshallbewings*. Если Вы все сделали правильно, и если первая написанная Вами миссия - это миссия приведенная в Примере № 1, то Вы в течении нескольких секунд должны получить три заданных там сообщения сообщения. Если же Вы пытаетесь оттестировать другую написанную Вами миссию - значить Вы знаете, что Вам следует ожидать от нее.

Для упрощения процесса тестирования и отладки миссий их разработчиками, в процессе их выполнения создается специальный файл, содержащий информацию по возникающим ошибкам, как в Вашем коде, так и в самой игре. Это файл *director.dmp* – он создается в основной папке игры, и помиллисекундно ведет журнал всего, что происходит с Вашей миссией. Он содержит всю возможную полезную информацию, включая сообщения об ошибках.

Пример-руководство по написанию "космических"('IN SP ACE') миссий

В этом разделе мы пройдем с Вами через весь процесс написания миссии, шаг за шагом, с подробным описанием всех ключевых моментов, и структурных элементов файла Миссии. Эта учебная Миссия основывается на старой доброй миссии из X -Tension и называется Docking Race Bet. Несмотря на то, что данный раздел - не самое удачно место для обучения работе в VS2005, мы все таки дадим Вам несколько полезных советов по работе с этой программой.

- Если у Вас пропал список возможных значений/пар аметров, Вы всегда можете вернуть его на экран, нажав Ctrl-Space.
- Файлы Миссий сами следят за правильностью своего форматирования.
- Для удобства просмотра Ваших Миссий, особенно их иерархии, Вы можете использовать метки -/+ в левой части экрана, для свертывания и развертывания Эпизодов и их элементов.
- Наведение указателя мыши к элементу подчеркнутому синим или красным вызовет подсказку, поясняющую возникшую проблему.
- Наведение указателя мыши на любой элемент вызовет подсказку - описание этого элемента.
- Если у Вас на экране отображен список возможных значений/параметров элемента - то перемещаясь по нему, Вы будете получать комментарий - описание текущего выбранного элемента списка.

Первый основной элемент файла - это его заголовок, в нашем случае состоящий из 3 строк.

Эта строка описывает версию .xml файла и кодировку, в которая была использована при записи информации в файл.

```
<?xml version="1.0" encoding="utf-8" ?>
```

Эта строка описывает стиль форматирования текста в самом XML файле .

```
<?xml-stylesheet href="director.xsl" type="text/xsl" ?>
```

Эта длинная строка описывает местоположение файла схемы, который используется для описания всех команд и структур наличест вующих в MD.

```
<director name="template" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="director.xsd">
```

Важно помнить, что атрибут описывающий наименование тега <director> никак не используется в игре, однако при просм отре XML файла в браузере это значение этого атрибута на самом верху страницы, в качестве ее заголовка. Это общее имя файла файла, но никоим образом не имя Эпизода(-ов).

Документация <documentation>

Содержание следующих нескольких строк говорит само за себя, и они не должны быть пустыми в любой официально выпущенной Миссии. Раздел `<documentation>` призван обеспечить тестеров и разработчиков информацией о том, кто написал эту М иссию, и как с ним связаться, для того чтобы дать обратную связь, и прокомментировать работу данной Миссии. Кроме того в ней содержится информация касательно версии, даты написания и статуса Миссии.

```
<documentation>
   <author name="Fred Bloggs" alias="Froggs" contact="froggs@egosoft.com"/>
   <content name="Docking Race Bet" description="Игрок получает предложение от
   пилота истребителя поставить деньги на то, что первый из них достигнет одной
   из торговых станций"/>
   <version number="1.0" date="2006-06-15" status="beta test"/>
</documentation>
```

В данном примере мы используем собственную внутреннюю нумерацию для Эпизодов внутри Миссии, и ставим номер в качестве префикса при именовании Эпизодов. Использование подобных префиксов не является обязательным, однако поступая так, мы снижаем риск совпадения наименования создаваемого Эпизода с наименованием Эпизодов в других файлах Миссий, написанных другими Вами, либо другими разработчиками. Критически важно использовать уникальные имена для Эпизодов, чтобы не происходило переопределение одних Эпизодов другими, и чтобы система Миссий была работоспособна! Потому соблюдение уникальности именования Эпизодов должно войти у Вас привычку, и система, подобная той, что приведена в данном примере - достаточно удобна и понятна, как при написании Миссии, так и при ее анализе

Эпизоды - <cues>

Следующая обязательная секция файла Миссий - это конечно `<cues>`. Как мы уже описывали ранее, как только Вы введете (выберете) тег `<cues>`, его закрывающая скобка `</cues>` должна тут же появиться в тексте файла Миссии. После нажатия Вами клавиши Энтер(return) курсор перейдет на следующую строку и сдвинется вправо, в позицию расположения подсекции. Прежде чем мы продолжим, хотелось бы напомнить, что внутри секции `<cues>` может содержаться неограниченное число подсекций вложенных `<cue>` и `<cues>`. Однако, сконцентрируемся на решении текущей задачи.

Эпизод - <cue>

Первый Эпизод(<cue>), который описывается в любой Миссии называется Эпизодом верхнего уровня, это означает что все другие секции <cue> и <cues> являются вложенными(подчиненными) и напрямую зависят от Условий Активации, Хронометража и Действий, описанных в этом главном Эпизоде. Фактически Эпизод верхнего уровня является триггером, определяющим поведение всей Миссии в целом.

```
<cues>
   <cue name="S01M25S00prepare">
```

Как мы уже неоднократно говорили, очень важно соблюсти уникальность наименования Эпизодов.

Теперь опишем другие атрибуты тега `<cue>`, которые задаются при написании Миссий.

`game=""` - опеределяет тип игры, в которой данная миссии будет доступна - к примеру: *all* (в любой), *plot* (сюжетная), *noplot* (несюжетная), *custom*(своя игра). (По умолчанию, если значение не задано, то подразумевается значение `"all"`).

`map=""` - определяет имя карты, на которой данная миссиия может быть выполнена.

`instantiate=""` - значение `"static"` определяет наличие только одной рабочей копии обрабатываемого Эпизода(-ов) . Оказывает влияние не только на Эпизод верхнего уровня. За более подробной информацией обратитесь к Приложению 1.

`library=""` - если задан - то относит данный Эпизод к *"библиотечным"*, что означает, что все Условия Активации, Хронометраж, и Действия могут быть использованы, только при ссылке на него с использованием атрибута `ref=""`.

`ref=""` - используется для ссылки на *"библиотечный"* Эпизод.

`comment=""` - используется для описания любого тега, начиная с `<cue>`, и служит для краткого информирования тестеров и разработчиков о назначении данной секции. Очень полезен при тестировании и отладке.

Более подробное описание использования атрибутов `library=""` и `ref=""` Вы найдете в Приложении 2: Библиотечные Эпизоды

Теперь рассмотрим три секции, определяющие любой Эпизод:

Условия Активации - <condition>

Для того, чтобы наша Миссия стала активной, необходимо соблюдение сразу нескольких условий.

```
1            <condition>
2               <check_all>
3                  <object_changed_sector comment="эта проверка осуществляется первой,
                   если условие не соблюдено, то остальные не будут проверены"/>
4                  <match_object class="fighter" comment="M3, M4 or M5"/>
5                  <check_value value="{player.age}" min="1m"/>
6                  <!--
7                  <check_age         min="10m"         chance="(72000/{player.time})+1"
                   comment="задает снижение шанса выполнения условия по времени игры,
                   со 100% в первые 10 минут до 1% после 20 часов"/>
8                  -->
9                  <object_has_equipment comment="если у нас установлен Стыковочный
                   Компьютер, мы можем принимать участие в гонках">
10                    <ware typename="SS_WARE_TECH241" min="1"/>
11                 </object_has_equipment>
12                 <check_value value="{player.money}"
                   min="{reward.money@veryeasy.XXXT}+500"/>
13                 <check_value value="{player.sector.race}"
                   exact="{lookup.race@argon}"/>
14              </check_all>
15           </condition>
```

Теперь разберем содержимое секции <condition> шаг за шагом:

1&15: Открывающие и закрывающие скобки тега <condition>, явно не требуют более подробного описания.

2&14: Открывающие и закрывающие скобки тега<check_all>, который позволяет разместить несколько Условий Активации, которые **должны быть соблюдены все**, для активации Эпизода.

3: Первое условие, соблюдение которого будет проверено, это `<object_changed_sector>`, это событийное условия, и выполняется при прибытии объекта в новый сектор, либо при стыковке/отстыковке с другим объектом. Если объект проверки явно не задан, то подразумевается текущий корабль Игрока.

4: Это условие проверяет принадлежность текущего корабля игрока к классу *"истребителей"*, *M3*, *M4* or *M5*.

5: В этом условии мы проверяем то, что время, прошедшее с начала игры как таковой ("возраст игрока") `{player.age}`, как минимум равно 1-й минуте.

6&8: Теги `<!-- -->` являются открывающей и закрывающей скобками комментариев, т.е. текст между ними не будет воспринят как текст Эпизода, и не будет обработан. Таким образом, используя их, Вы в любой момент можете временно исключить часть кода Ваших Миссий. Так же Вы можете использовать их для более подробного описания Миссий.

7: Это альтернативное условие, в настоящий момент „*"закомментировано"*, и вместо него, в отладочных целях используется условие в предыдущей строке. В игре данное условие будет вести себя так, как описано в комментарии к нему.

9-11: В этом условии проверяется наличии на корабле Игрока *Стыковочного Компьютера*. Подсекция `<ware>` служит для задания конкретного типа товара, наличие которого проверяется, в данном случае это *SS_WARE_TECH241*. Вы можете выбрать его из списка возможных значений для данной секции. Как уже говорилось, хорошим тоном является использование комментариев, для описания каждого шага Миссии. Обратите внимание, что в данной подсекции не задан атрибут `object=""`, что означает, что оно относится к текущему кораблю Игрока. Держите эту особенность в голове, как при написании, так и при анализе Миссий. Здесь Вы можете задать неограниченное число подсекций `<ware>`, для проверки наличия необходимого Вам списка оборудования и товаров.

12: В данном условии осуществляется проверка того, что сумма денег на счету Игрока `{player.money}` как минимум равно сумме вознаграждения за миссию + 500Cr. Эта проверка осуществляется на случай проигрыша Игрока. Предустановленные значения вознаграждений за разные типы и уровень сложности Миссий Вы сможете посмотреть, открыв файл *director.html*.

13: И последнее условие, которое будет проверено только при соблюдении всех предыдущих - это сравнение двух переменных. Первая - это `{player.sector.race}`, которая содержит идентификатор (*ID code)*, используемый игрой для описания расы-владельца сектора, в котором находится Игрок. Вторая - `{lookup.race@argon}`, которая содержит идентификатор (*ID code*) для расы *Аргон*(*Argon*). При точном совпадении значений этих переменных условие будет выполнено. Т.е. говоря простым языком, здесь проверяется то, что текущим сектором владеет раса *Аргон*.

Вот собственно и все те условия, которые должны быть соблюдены для активации .

Не забывайте, что использование `<check_all>` требует соблюдения **всех** перечисленных условий.

Совет: Если Вам нужно проверить соблюдение нескольких обязательных условий, и хотя бы одного из возможных, Вы должны использовать вложенный тег `<check_any>` внутри `<check_all>`. Если же Вам необходимо проверить соблюдение только одного из блоков, содержащих более одного обязательного условия, вам нужно использовать на верхнем уровне тег `<check_any>` содержащий вложенные `<check_all>`.

Хронометраж - <timing>

В этом Эпизоде верхнего уровня секции `<timing>` проста до невозможности:

```
<timing>
   <time min="3s" max="5s"/>
</timing>
```

Данный хронометраж означает, что в период между 3-мя и 5-ю минутами от факта соблюдения Условия Активации Эпизод должен быть активирован.

Действия - <action>

Так как мы уже изучили секцию Условий Активации(`<condition>`), то от нее рукой подать до секции Действий (`<action>`), к изучению которой мы сейчас и приступим.

```
1  <action>
2     <do_all>
3        <find_gate nearest="1" name="this.gate">
4           <distance max="10km"/>
5        </find_gate>
6        <find_station name="this.finish" class="trade" dockingallowed="1"
         findobject="{player.ship}" max="1" typename="SS_DOCK_A_TRADE"/>
7        <set_value name="this.reward"
         exact="({reward.money@veryeasy.XXXT}/100)*100" comment="излишние
         /100 *100 служат для получения округленного значение : -)"/>
8     </do_all>
9  </action>
```

1&9: Это открывающая и закрывающая скобки тега <action>.

2&8: Тег `<do_all>` задает подсекцию, содержащую список действий, которые в подлежат последовательному и обязательному выполнению, в порядке их следования в подсекции.

3-5: Команда <find_gate> осуществляет поиск ближайших Врат, причем, учитывая то, что задана подсекция `<distance>`, поиск ведется в сфере радиусом 10 km. В случае нахождения Врат, результат будет записан в переменную с именем `"this.gate"`. Использование именно такой формы именования достаточно критично, т.к. если б ы мы задали в качестве имени переменной только `"gate"`, то была бы создана глобальная переменная, доступная в любом месте игры и во всех файлах MD. В данном же случае, мы хотели создать локальную

переменную, которая была бы доступна только в пределах текущ его Эпизода(<cue>). Именное поэтому был использован префикс "this."

6: Командой <find_station> мы осуществляем поиск станции, которая будет фигурировать в Эпизоде в виде переменной с именем "this.finish" и должна принадлежать к классу "trade", т.е. являться *Торговым Доком*. А еще точнее - должна принадлежать конкретному типу торговых доков - *SS_DOCK_A_TRADE*, т.е. быть *Federal Argon Trading Dock*. Кроме того, текущий корабль Игрока, а именно объект {player.ship}, должен иметь возможность осуществить стыковк у с ней.

7: Команда <set_value> служит для задания значения или диапазона значений, в нашем случае - в переменную с именем "this.reward". Как и в случае с ранее описанными переменными, эта переменная является локальной.

Присваемое значение вычисляется по формуле ({reward.money@veryeasy.XXXT})/100)*100. Следует помнить, что все вычисления в MD осуществляются только с целыми числами. Выражения, заключенные в скобки, вычисляются в первую очередь, с учетом вложенности скобок (кроме скобок, используемых для задания переменных { }). Таким образом, в нашем примере, мы используем предопределенное значение вознаграждения для миссии с самой низкой сложностью(very easy), принадлежащей к классу *XXXT*. Данное значение делится на 100, а затем умножается на 100, как сл едует из комментария, это делается для получения округленного значения.

Вот собственно и весь Эпизод верхнего уровня в нашей Миссии. В этом Эпизоде мы задаем Условия Активации Миссии, ее Хронометраж, а так же определяем ключевые объекты, с которыми связа на Миссия - Врата и станцию, которая должна послужить финишем для нашей гонки. Кроме того, мы определяем размер вознаграждения за победу в гонке.

В следующем Эпизоде, который является вложенным эпизодом в уже рассмотренный нами, будет проделаны подготовите льные действия, которые необходимо выполнить перед тем, как мы сможем дать старт нашей гонке. Объектами воздействия данного Эпизода будут служить объекты, определенные в родительском Эпизоде, т.е. Врата и станция, если конечно они были . Так же мы сможем у видеть, что же произойдет, если они не были найдены. А так же получим наглядный пример использования вложенных условий в данном Эпизоде.

Во-первых, каким образом создать Эпизод в качестве вложенного Эпизода? Как уже было описано ранее в данном документе, Эпизоды могут быть вложены в другие Эпизоды. Для этого мы должны создать подсекцию, ограниченную тегом <cues> до закрытия текущего Эпизода (секции и тега <cue>). Эта подсекция <cues> должна быть расположена сразу за закрывающей скобкой секции Действий, т. е. сразу за тегом </action>, в текущем Эпизоде. Выглядеть это должно так:

```
   </action>
   <cues>
     <cue name="blah">
```

В данном примере в подсекции Эпизодов <cues> описан как минимум один суб-эпизод <cue>. При написании миссий вы можете использовать столько вложенных Эпизодов, и с таким уровнем вложенности, сколько и какой вам потребуется.

Условия Активации <condition>

Итак, приступим к рассмотрению следующего Эпизода, и начнем мы с Условий …

```
1            <cues>
2               <cue name="reset">
3                  <condition>
4                     <check_all>
5                        <cue_is_complete cue="S01M25S00prepare"/>
6                        <check_any>
```

```
7                           <object_exists negate="1" object="S01M25S00prepare.gate"/>
8                           <object_exists negate="1"
                            object="S01M25S00prepare.finish"/>
9                        </check_any>
10                    </check_all>
11                 </condition>
```

1: Как мы и говорили, мы создаем секцию вложенных Эпизодов `<cues>`, следовательно следующим тегом должен быть `<cue>`.
2: Как мы и обещали … Мы используем для этого Эпизода имя `"reset"`, потому что, именно в нем осуществляется сброс параметров.
3&11: Открывающая и закрывающая скобки подсекции Условий Активации(`<condition>`).
4&10: `<check_all>` - все указанные условия должны быть соблюдены.
5: Первое условие, соблюдение которого будет проверено - это то, что Эпизод с именем `"S01M25S00prepare"`, в нашем случае это Эпизод верхнего уровня верхнего уровня, должен находиться в статусе *"завершен"*, т.е. все Условия Активации соблюдены, и все Действия - выполнены.
6&9: Вложенная, в текущую секцию условий `<check_all>`, подсекция `<check_any>` содержит список условий для которых нам достаточно соблюдения любого из них, чтобы считать, что вся данная подсекция условий - соблюдена. Учитывая текущую вложенность их можно описать выражением *"и если любое из"*.
7&8: Нам необходимо убедиться, что из искомых нами, с помощью команд *"find"*, объектов (Врата и станция), обнаружены и существуют оба. Таким образом нам надо проверить, а есть ли среди объектов `.gate` or the `.finish` хотя бы один не существующий. Если оба эти объекта существуют, итоговое Условие Активации данного Эпизода не будет.

Вас может удивить наличие префикса *S01M25S00prepare*, хотя ранее мы для этих переменных использовали префикс *this*; префикс *this.blah* используется для описания локальных переменных внутри того Эпизода, в котором они были объявлены(заданы). Если существует необходимость обратиться к ним из другого Эпизода, то вместо префикса *"this"* следует использовать имя Эпизода, в котором они были объявлены(заданы). В случае же обращения к переменным Эпизода, являющегося родительским, по отношению к текущему, возможно использование префикса *"parent"*, т.е. переменная будет записана как *parent.blah*.

Хронометраж - <timing> и Действия - <action>

После того, как мы осуществили проверку, и убедились, что родительский Эпизод завершен, но, либо Врата, либо станция, не были найдены, либо не существуют, нам необходимо выполнить некие действия. Собственно, необходимо задать последствия, отсутствия Врат или станции. Если мы убедились в том, что один из объектов не существует, то наша миссия не сможет отработать корректно.

```
                  <timing>
                     <time min="1s"/>
                  </timing>
```

Снова простая и понятная секция Хронометража. По истечении одной секунды после соблюдения Условий Активации, Действия данного Эпизода должны быть выполнены.

```
1                 <action>
2                    <reset_cue cue="S01M25S00prepare"/>
3                 </action>
4              </cue>
```

Такая маленькая секция Действий, но очень важная для Миссии в целом. В данном случае, мы прерываем выполнение текущей Миссии, потому что один из ключевых, для ее успешного выполнения, объектов не существует. Следовательно на необходимо отменить старт Миссии в целом и ожидать соблюдения Условий Активации при следующей смене

Игроком текущего сектора. Это означает, что проверка на наличие и станции и Врат будет осуществлена вновь, до тех пор, пока оба эти объекта не будут найдены.

То что закрывающая скобка `</cue>` следует сразу за секций Действий, означает что в данном Эпизоде вложенные Эпизоды отсутствуют.

Следующий Эпизод может показаться Вам слишком хитроумно написанным, особенно с учетом того, что он довольно велик, однако мы постараемся разбить его описание на части, легко поддающиеся осмыслению.

```
1              <cue name="selectOpponent" comment="выбор корабля-соперника, основанный
               на классе корабля">
2                 <condition>
3                    <cue_is_complete cue="S01M25S00prepare"/>
4                 </condition>
```

1: Задание имени Эпизода, с комментарием, в котором содержится краткое описание назначения данного .

2-3: Условием Активации данного Эпизода является просто переход родительского Эпизода (*S01M25S00prepare*) в состояние Выполнен, т.е. все условия в нем соблюдены и действия - выполнены.

Секция Действий `<action>` данного Эпизода больше тех, что мы рассмотрели ранее, и потому, приготовьтесь к тому, что разобраться в ней может быть непросто.

```
1                 <action>
2                    <do_choose>
3                       <do_when value="{player.ship.class}"
                        exact="{lookup.class@m3}">
4                          <set_value name="selectOpponent.typename"
                           exact="{lookup.type@{random.type@SS_SH_A_M3|SS_SH_A_M3_1|SS
                           _SH_A_M3_2|SS_SH_A_M3_3}}"/>
5                       </do_when>
6                       <do_when value="{player.ship.class}"
                        exact="{lookup.class@m4}">
7                          <set_value name="selectOpponent.typename"
                           exact="{lookup.type@{random.type@SS_SH_A_M4|SS_SH_A_M4_1|SS
                           _SH_A_M4_2|SS_SH_A_M4_3}}"/>
8                       </do_when>
9                       <do_otherwise comment="not M3 and not M4 so I'm expecting M5">
10                         <set_value name="selectOpponent.typename"
                           exact="{lookup.type@{random.type@SS_SH_A_M5|SS_SH_A_M5_1|SS
                           _SH_A_M5_2|SS_SH_A_M5_3}}"/>
11                      </do_otherwise>
12                   </do_choose>
13                </action>
14             </cue>
```

1&13: Скобки тега `<action>`, там где он и и должны находиться.

2&12: Все действия выполняемые данным Эпизодом регулируются тегом `<do_choose>`, который позволяет Вам задать дополнительные *"условия"* для выполнения конкретных действия. Фактически реализована следующая логика „если a=b тогда выполняем c, иначе выполняем d‟ . Теперь рассмотрим, как же это работает на конкретном примере.

3&5: Первое секция `<do_when>`, которая ограничивает действия, выполняемые при соблюдении *"условия"*, когда внутренний идентификатор класса корабля Игрока будет совпадать с возвращаемым значением для кораблей *M3*.

4: Только в случае, если корабль Игрока относится к классу *M3*, будет выбран тип корабля-соперника, соответствующий возвращаемому значению (внутреннему идентификатору). Обратите внимание, на символ „|‟ , который использован при задании переменной random.type, в данном применении он означает *"or(или)"*, и в результате будет случайно выбрана одна из 4 модификаций корабля *Nova*.

6&8: Аналогично **3&5**, здесь мы проверяем соответствие внутреннего идентификатора класса корабля Игрока возвращаемому значению для класса *M4*.

7: Аналогично 4, только при принадлежности корабля Игрока к классу *M4* выбор осуществляется среди 4 модификаций *Argon M4 Buster*.

9&11: В отличии от ранее использованного тега `<do_when>`, здесь мы использовали `<do_otherwise>`, которое ограничивает действия, выполняемые при несоблюдении всех ранее описанных в секции `<do_choose>` *"условий"*. Т.е. если корабль Игрока не принадлежит ни к классу *M3*, ни к классу *M4*, он должен принадлежать к классу *M5* (ранее мы ограничили классы корабля Игрока принадлежностью его к типу *Истребителей* в Условии Активации родительского .

10: В данном случае, в качестве соперника, будет выбрана одна из четырех модификаций *Argon Discoverer* (*M5*) .

Примечание: Во всех секциях `<set_value>` внутренний идентификатор (возвращаемое значение) типа корабля присваивается переменной `“selectOpponent.typename”`. Эта переменная может быть использована для создания соперника в следующем.

14: Мы закрываем данный Эпизод `"selectOpponent"`, он не имеет никаких вложенных Эпизодов.

Следующий Эпизод находится на том же уровне вложенности, что и предыдущий, т.е. он является вложенным в *S01M25S00prepare*.

```
<cue name="createopponent">
```

Как и во всех ранее рассмотренных Эпизодах, начнем ео изучение (и написание) с секции Условий Активации, задаваемой тегом `<condition>`. Разве что стоит отметить, что в данном Эпизоде отсутствует секция Хронометража, что означает, что Эпизод будет активирован и секция Действий будет выполнена сразу по соблюдению Условий Активации.

```
1                  <condition>
2                     <check_all>
3                        <cue_is_complete cue="selectOpponent"/>
4                        <cue_is_complete cue="S01M25S00prepare"/>
5                        <object_exists object="S01M25S00prepare.gate"/>
6                        <object_exists object="S01M25S00prepare.finish"/>
7                     </check_all>
8                  </condition>
```

Условия Активации данного Эпизода в общем случае подобны рассмотренному ранее Эпизоду с именем `“reset”`.

1&8: Скобки секции Условий Активации

2&7: Все условия заключены в скобки тега `<check_all>`, следовательно мы ожидаем соблюдения всех условий.

3: Эпизод selectOpponent должен иметь статус Выполнен.

4: Эпизод *S01M25S00prepare*, т.е. Эпизод верхнего уровня, должен иметь статус Выполнен.

5: Объект - Врата, поиск которого мы осуществляли в родительском Эпизоде, должен быть найден и существовать.

6: Объект - станция, поиск которого мы осуществляли в родительском Эпизоде, должен быть найден и существовать. .

Итак, если все условия соблюдены, соперника выбран (тип корабля и его модификация), Врата и станция обнаружены и существуют, следовательно мы можем приступить к выполнению действий, о которых говорит само название данного Эпизода, т.е. перейти к созданию корабля-соперника(оппонента) для Игрока в наших гонках.

Внимание! Секция Действий этого Эпизода довольно-таки обширна, и потому, если вы решили перекусить, выпить кофе(колы, пива, ...) или съесть попкорна(чипсов, бутерброд, выпить пива ....) - озаботьтесь этим сейчас.

Как мы уже предупреждали ранее, секции Хронометража в данном Эпизоде нет, и потмоу мы приступаем к рассмотрению секции Действий.

```
1                  <action>
2                     <do_all>
3                        <create_ship name="this.opponent"
                         typename="{value@selectOpponent.typename}" racelogic="0"
                         race="argon" class="{player.ship.class}" highlight="1">
4                           <position min="5km" max="6km"
                            object="S01M25S00prepare.gate"/>
5                           <equipment loadout="default"/>
6                           <command command="follow" commandobject="{player.ship}"/>
7                           <pilot name="{random.pilot.argon}" race="argon"/>
8                        </create_ship>
9                        <set_value name="this.tunings"
                         exact="{player.ship.maxspeed}/({object.basespeed@this.opponent
                         }/10)-10"/>
10                       <add_equipment object="this.opponent">
11                          <ware typename="SS_WARE_TECH213" exact="-100"
                            comment="removing all engine tunings as a workaround "/>
12                          <ware typename="SS_WARE_TECH213" min="{value@this.tunings}-
                            1" max="{value@this.tunings}+1" comment="Engine Tunings"/>
13                       </add_equipment>
14                       <do_if negate="1" value="{object.class@this.opponent}"
                         exact="{lookup.class@m5}" comment="на корабли  класса M5
                         невозможно установить Прыжковый двигатель, поэтому мы даже не
                         будем пытаться сделать это">
15                          <add_cargo object="this.opponent">
16                             <ware typename="SS_WARE_ENERGY" min="15" max="50"
                               comment="Energy Cells"/>
17                          </add_cargo>
18                          <add_equipment object="this.opponent">
19                             <ware typename="SS_WARE_WARPING" exact="1"
                               comment="Jumpdrive"/>
20                          </add_equipment>
21                       </do_if>
22                       <ask_question name="bet" author="{object.pilot@this.opponent}"
                         text="{1081,1}"/>
23                    </do_all>
24                 </action>
```

Итак приступим, начнем с самого простого:

1&24: Скобки секции Действий (тег `<action>`).

2&23: Все действия заключены в скобки тега `<do_all>`, что, как мы надеемся вы еще помните, означает, что все действия, описанные в строках **3-22**, должны быть выполнены.

3&8: Первое, что мы должны сделать - это, используя команду `<create_ship>` и используя необходимые атрибуты, создать соперника для Игрока. В качестве атрибута, в который будет записан созданный объект, мы используем наименование `"this.opponent"`. Так же мы задаем конкретный тип корабля, используя внутренний идентификатор для модели. В данном случае, мы используем значение, полученное нами в предыдущем Эпизоде, а именно в *selectOpponent*. Расовая логика при создании отключена, т.е. *Racelogic* установлено в 0, что означает, что после создания он не будет действовать как обычный расовый корабль, в данном случае - *Аргонский*. Так же при создании атрибут `highlight="1"`, что позволяет выделить его на карте сектора.

4: Созданный корабль будет помещен в позицию на расстоянии в 5 - 6 км от объекта Врат, заданного в родительском Эпизоде. Обратите внимание что мы используем не переменную *this.gate*, как она использовалась в нем, а обращаемся к ней используя префикс с именем Эпизода, в котором она была определена, т.е. *S01M25S00prepare.gate*, таким образом, мы используем результат выполнения предыдущего кода.

5: Теперь приступим к установке оборудования на корабль-соперник, при этом мы можем использовать некие предопределенные наборы комплектации такого оборудования,

в нашем случае использован набор `"default"`, а затем он модифицируется используя теги `<ware>`.
Существует три различных типа комплектации (`"loadout"`):

- минимальный - minimum :
  - один щит на 1MW;
  - одна турель заполняется лазерами - обычно главная;
  - ракет нет;
  - только базовые апгрейды;
  - улучшения скорости, управляемости и грузового отсека не устанавливаются.
- типовой - default :
  - лучшие щиты, иногда возможна установка и чего-то более слабого, хотя и с минимальной вероятностью;
  - лучшее вооружение, иногда возможна установка и чего-то более слабого, хотя и с минимальной вероятностью;
  - ракеты - есть, но не самые лучшие;
  - небольшое количество апгрейдов и улучшений.
- максимальный - maximum :
  - лучшие щиты;
  - лучшее вооружение, причем возможно установка чего-либо необычного;
  - лучшие ракеты;
  - а также апгрейды и улучшения, подобные устанавливаемым в типовой комплектации.

6: И в качестве одной из составляющих секции создания корабля (`<create_ship>`) мы даем команду кораблю приблизиться к кораблю Игрока, используя команду *“follow”*.
7: Используя команду `<pilot>` мы задаем имя пилота созданного корабля, используя специальную переменную, которая возвращает одно из произвольно выбранных имен пилотов, принадлежащих к расе *Аргон*, хотя могли бы и воспользоваться текстовой строкой либо ссылкой на текстовую строку в текстовом файле. Так же мы задаем *Аргон* в качестве расы пилота. Обратите внимание, что это расовая принадлежность пилота, а не корабля..
9: В данной строке мы присваиваем переменной `{this.tunings}` результат вычисления выражения, передаваемого через атрибут `value=""`. В данном случае - это максимальная скорость корабля Игрока, разделенная на десятую часть скорости корабля-соперника, и еще уменьшенная на 10. Это число, в дальнейшем, будет использовано нами как множитель, при установки улучшения двигательной установки на корабль-соперник.
10&13: Скобки секции установки оборудования, описываемой тегом `<add_equipment>`. Данная команда(секция) служит для установки необходимого оборудования на корабль-соперник, используя значение, вычисленное в строке **12**.
11: Данная команда в секции `<add_equipment>` удаляет 100 единиц улучшений двигательной установки на корабле-сопернике, с последующей установкой необходимого числа улучшений в следующей строке. Это делается для того, чтобы, теоретически, удалить все возможно ранее установленные улучшения, т.е. получить корабль с исходными характеристиками скорости.
12: Эта же команда, в секции `<add_equipment>`, устанавливает произвольное количество улучшений двигательной установки, лежащее в диапазоне между `{value@this.tunings}-1` и `{value@this.tunings}+1`.
14&21:Здесь мы впервые используем команду `<do_if>`. Атрибут `negate=""` используется для того, чтобы определить, при каком значении заданного условия, будет исполняться секция вложенных команд. Если не установлен - то при соблюдении условия, если установлен в 1 - то при несоблюдении. В данном случае, вложенные действия будут выполнены, если корабль не принадлежит к классу *M5*. Как следует из комментариев - на корабли такого класса невозможно установить *Прыжковый двигатель*, и потому нет смысла пытаться

сделать это, и следовательно действия, заключенные в скобки `<do_if>` будут выполнены только для кораблей класса *M3* или *M4*.
15-17: В данной строке мы командой `<add_cargo>` добавляем в грузовой отсек корабля-соперника от 15 до 50 батареек.
18-20: В данной секции установки оборудования (`<add_equipment>`) мы устанавливаем *Прыжковый двигатель* на корабль-соперник. Это оборудование не используется в гонке, однако позволяет сопернику "исчезнуть", как и предусмотрено.
22: Завершающее действие в данном Эпизоде - это команда `<ask_question>`. Данному запросу присваивается имя, для того чтобы использовать данные, полученные в результате его обработки в других Эпизодах данной Миссии. Запрос оформляется в виде входящего сообщения Игроку, следовательно оно должно иметь автора, в данном случае это имя пилота созданного нами корабля-соперника *this.opponent*. Далее следует текст самого запроса. В данном случае мы использовали ссылку на текстовую строку из текстового файла, хотя это может быть и просто текстовая строка. Текст, используемый в стандартных(официальных) миссиях обычно помещается в текстовый файл, поэтому ссылка на текстовую строку из такого файла - предпочтительнее, чем прямой набор текста в файле Миссии. Это позволяет легко решать вопросы локализации текста Миссий. К тому же мы рекомендовали бы Вам писать, по-возможности, короткий текст

Уф-ф! Ну вот мы наконец и создали этого соперника, и даже задали от его имени вопрос. Для лучшего понимания, что же будет происходить дальше, имеет смысл взглянуть на текст сообщения, которое мы отправляем Игроку:

```
Говорит Аргонский пилот {object.pilot@createopponent.opponent}. По хоже у тебе довольно-таки быстрый кораблик!\nГотов побиться об заклад, например на сумму в {value@S01M25S00prepare.reward} кредитов, что я пристыкуюсь к вон той, крупной станции {object.name@S01M25S00prepare.finish}, в центре сектора, раньше тебя?[center ][select value='yes']Я тебя сделаю![/select][/center][center][select value='no']Вот еще, гонки тут устраивать![/select][/center]
```

В тексте сообщения использованы некоторые переменные.

- `{object.pilot@createopponent.opponent}` - это имя пилота созданного нами корабля-соперника *this.opponent*;снова мы использовали имя Эпизода вместо использования префикса *"this."*, следовательно проблем с нахождение значения этой переменной у движка МД не возникнет.
- `{value@S01M25S00prepare.reward}` - была определена ранее, в Эпизоде верхнего уровня
- `{object.name@S01M25S00prepare.finish}` - так же была определена ранее, в Эпизоде верхнего уровня. Опять же в том Эпизоде она определялась как переменная `"this.finish"`.

Использование команды `<ask_question>` определяет необходимость наличия в тексте запроса как минимум одного вложенного Эпизода, а чаще всего - двух, по одному на каждый вариант ответ, в данном случае - один на положительный - *"да(yes)"* и один на отрицательный - *"нет(no)"*. В каждом из этих вложенных Эпизодов мы должны проверить, какой вариант ответа был выбран, и выполнить действия, базируясь на выбранном варианте ответа. Ну что ж, давайте посмотрим, как мы будем разбирать ответ на запрос сформированный от имени соперника (`"this.opponent"`).

Так как вложенные Эпизоды, в которых разбирается ответ на сформированный запрос к Игроку должны быть расположены в Эпизоде ,в котором он был задан, то следующая строка, идущая за закрывающей скобкой блока Действий (`</action>`) должна содержать открывающую скобку блока вложенных Эпизодов (`<cues>`). А у же за ней - и первая строчка одного из вложенных Эпизодов.

```
                    </action>
                    <cues>
                       <cue name="noBet">
```

Как и ожидалось, Условием Активации данного Эпизода будет проверка состояния сформированного нами запроса к Игроку, а именно то, что ответ был дан, выбор Игроком - сделан, и мы проверяем, какой именно.

```
<condition>
   <question_answered question="bet" answer="no" />
</condition>
```

Обратите внимание, что атрибут `question=""` имеет значение `"bet"`, т.е. имя, которое мы ранее присвоили сво ему запросу к Игроку в родительском . Таким образов в данном Условии Активации проверяется ответ Игрока на наш запрос `"bet"`, и Эпизод активируется, если Игрок выбрал вариант *"нет"* т.е. отказался делать ставку. При этом будут выполнены следующие действия:

```
<action>
   <do_all>
      <incoming_message
      author="{object.pilot@createopponent.opponent}"
      text="{1081,9}"/>
      <cancel_cue cue="closetoplayer"/>
   </do_all>
</action>
```

`text="{1081,9}"` is `"Я должен был сразу догадаться, что тот, кто летает на такой калоше не достоин участия в такой гонке. Сгинь с глаз моих!"`

Этот эпизод совершенно незамысловат, особенно в сравнении с предыдущим; фактически происходит отсылка сообщения игроку (его текст приведен выше), а так осуществляется прерывание выполнения(отмену) Эпизод а, который следует за этим. Кстати, не забудьте, что прерывание/отмена Эпизода влечет за собой и прерывание/отмену всех вложенных эпизодов. В данном случае Эпизод, который будет прерван/отменен - это Эпизод, обрабатывающий ответ *"да(yes)"* на наш запрос. Таким образом, ответ *"нет(no)"* фактически завершает исполнение всей Миссии в целом, как и нам и необходимо.

Прежде чем мы перейдем к рассмотрению Эпизода, который связан с ответ ом *"да(yes)"* на наш запрос, нам осталось рассмотреть еще один, совсем небольшой Эпизод.

```
<cues>
   <cue ref="clean up">
      <params>
         <param name="time" value="10s"/>
      </params>
   </cue>
</cues>
```

Этот очень маленький Эпизод, как и положено, заключен в ск обки вложенных Эпизодов(<cues>), потому что является вложенным в Эпизод `"no bet"`, и находится сразу за закрывающей скобкой секции Действий данного Эпизода (`</action>`).

Как мы уже упоминали ранее, использование атрибута `ref=""` означает ссылку на библиотечный Эпизод. Следовательно, далее в нашем файле Миссии должен быть описан библиотечный Эпизод, с именем `"clean up"`. Этот библиотечный Эпизод `"clean up"` предназначен для аккуратного удаления из игры нежелательного для Игрока соперника, в случае, если Игрок отказался от участия в гонке (ответил *"нет(no)"*). Более подробно этот процесс мы с Вами рассмотрим позднее, когда доберемся до этого библиотечного Эпизода `"clean up"`. Назначение секции, ограниченной тегами `<params>` и `</param>` - передача заданных параметров в библиотечный Эпизод. При каждом использовании Эпизода `"clean up"`, он может быть вызван с различными значениями параметров, и действовать по разному, так как нам надо. Более подробное описание использования и работы параметров(`<params>`) сделано в Приложении 2. Библиотечные Эпизоды.

Следующий эпизод, как и Эпизод `"no bet"` является вложенным в Эпизод `"createOpponent"`, и потому, мы вначале должны закрыть скобку Эпизода `"no bet"`, а уже потом приступить к новому Эпизоду:

```
                    </cue>
                    <cue name="closetoplayer">
```

Это именно тот Эпизод, прерывание/отмену которого, вместе со всеми вложенными в него Эпизодами, мы рассмотрели при обработке отказа Игрока от участия в гонке.

Теперь мы должны быть оптимистами, и рассмотреть вариант , когда Игрок согласился сделать ставку. Следовательно, по аналогии с Эпизодом „no bet " , Условием Активации будет ответ *"да(yes)"*.

```
                    <condition>
                       <question_answered question="bet" answer="yes" />
                    </condition>
```

В отличии от Эпизода `"no bet"` , данный Эпизод содержит секцию Хронометража (`<timing>`) и фактически начинает выполнятся в течении 1-3 секунд после соблюдения Условий Активации:

```
                    <timing>
                       <time min="1s" max="3s"/>
                    </timing>
```

И, наконец, сами действия, выполняемые в том Эпизоде:

```
                    <action>
                       <do_all>
1                         <incoming_message
                          author="{object.pilot@createopponent.opponent}"
                          text="{1081,2}" popup="1"/>
2                         <create_object name="this.marker"
                          typename="SS_SPECIAL_MARKER" class="special">
                             <position min="3km" max="5km" object="player.ship}
                             "/>
                          </create_object>
3                         <set_target object="this.marker"/>
4                         <set_command object="createopponent.opponent"
                          command="moveposition" commandobject="this.marker">
                             <position object="this.marker"/>
                          </set_command>
                       </do_all>
                    </action>
```

В данной секции у нас описано несколько действий, заключенных в скобки обязательного выполнения `<do_all>`. Поэтому в данном случае мы пронумеруем (для своего удобства) только строки, связанные непосредственно с этими действиями, исключив из рассмотрения все остальные, с которыми мы уже должны были хорошо освоится (к примеру: `<cues>`, `<cue>`, `<condition>`, `<timing>`, `<action>` и `<do_all/any>`).

1: Первым делом, игроку будет отправлено сообщение от имени соперника, это сообщение буде всплывающим и сопровождаться звуковым сигналом, как входящее сообщение.

Вот текст этого сообщения `“Отлично! Я переключаю свой опознаватель друг/враг, и буду отображен на вашем радаре красной точкой. Удобно, не правда ли?! Только чур, не надо в меня стрелять, ведь я не враг! Пожалуйста, подлетите к отмеченной маркером точке пространства. Вы должны быть на расстоянии не более чем 200m от метки!”`

2: Далее, мы создаем метку, которая отмечает старт овую позицию наших гонок, и даем ей имя `"this.marker"`.

Не забудьте: если нам понадобиться обратиться к этому объекту из других Эпизодов, то обращение будет выглядеть как `"closetoplayer.marker"`.

Так же, мы задаем данному объекту выбранный тип и класс `"special"`. Внутри секции `<create_object>` мы так же задаем координаты, по которым должен быть создана метка, в данном случае, она должна находиться на расстоянии от 3 -х до 5-ти километров от корабля Игрока.

3: Теперь, используя тег `<set_target>` мы задаем указанную метку в качестве цели корабля Игрока. Заметьте, что в связи с тем, что мы используем эту переменную внутри Эпизода, в котором она была создана, то используется укороченное обращение `"this.marker"`.

4: Далее, используя тег `<set_command>`, мы даем команду кораблю-сопернику лететь к созданной метке. Использование метки в качестве объекта, относительно которого выполняется команда, а так же задание ее внутри секции, в качестве атрибута `<position>`, гарантирует, что корабль-соперник прибудет в заданную точку

Итак, резюмируем, что у нас уже сделано в данной Миссии на данный момент:

- Мы идентифицировали станцию(финишная позиция нашей гонки) и Врата (ориентировочная стартовая позиция нашей гонки).
- Если оба объекта найдены и существуют, тогда создается соперник, имеющий скорость близкую к скорости корабля Игрока, и им было инициировано предложение провести гонку до выбранной станции.
- Если Игрок говорит *"нет"*, соперник немножко унижает Игрока и исчезает
- Если Игрок говорит *"да"*, то создается метка стартовой позиции. Как уже говорилось ранее, на каждый вариант ответа Игрока был создан отдельный вложенный Эпизод, активация которого отменяет другой, и потом если получен ответ *"да"*,то стартует выполнение последовательности вложенных эпизодов, если *"нет"* то последовательность вложенных эпизодов будет отменена/прекращена.

Так как мы с Вами считаем что у нас идет светлая полоса в нашей жизни, то мы можем приступить к рассмотрению следующих вложенных Эпизодов, и увидеть, что же должно происходить после того, как дан старт гонке.

Так как мы с Вами уже не новички, то вполне можем рассмотреть следующий эпизод целиком:

```
<cues>
   <cue name="coward">
      <condition>
         <check_all>
            <object_changed_sector/>
         </check_all>
      </condition>
      <timing>
         <time min="1s"/>
      </timing>
      <action>
         <do_all>
            <incoming_message
            author="{object.pilot@createopponent.opponent}"
            text="{1081,114}" popup="1"/>
            <destroy_object object="closetoplayer.marker"/>
            <cancel_cue cue="inposition"/>
         </do_all>
      </action>
```

Таким образом, на примере данного Эпизода мы с Вами можем видеть структуру Эпизода целиком, с секциями Условий Активации, Хронометража и Действий.

В секции Условий Активации мы использовали конструкцию `<check_all>`, хотя он там фактически и не нужна, т.к. условие всего одно. Однако смысл в этом есть, так как таким образом мы подготовились к добавлению других условий, если потребуется. Условие

<object_changed_sector> по умолчанию работает с объектом - корабль игрока, если другой объект не задан явно. Т.е. если Игрок покинул текущий сектор по какой-либо причине, действие будет активировано с задержкой в одну секунду, как это определено в секции Хронометража.
Секция Действий содержит всего три элемента. Первая команда отправляет от имени соперника сообщение Игроку, следующего содержания "Трус! Ты покинул сектор. Пари разорвано!"
Затем, используя <destroy_object>, удаляем метку, которая должна была отмечать место старта предполагаемой гонки (обратите внимание на использование в качестве префикса имени Эпизода).
И наконец, прерываем/отменяем Эпизод "inposition", вместе с его вложенными Эпизодами.

Вас наверное немного смущает, то, что Миссия продолжает работать при наличии такого числа отмененных/прерванных или перезапущенных Эпизодов. Однако просто следует помнить, что движок игры ежесекундно сканирует все XML файлы Миссий. Сканирование проходит через все Эпизоды, с Эпизодов верхнего уровня до самого глубоко вложенного Эпизода, и для каждого Эпизода выполняется проверка Условий Активации. Если Условия Активации не были соблюдены, то проверка продолжается дальше, и так до конца файла Миссии. Это означает, что если вы не отменили/прервали Эпизод (и вложенные в него), исполнение которого больше не требуется, его Условия Активации будут проверяться на соблюдение, и для движка игры Миссия будет активной и продолжаться, хотя Вы будете уверены в обратном.

Следующий Эпизод является вложенным в Эпизод "coward". Еще один малыш.

```
1                                <cues>
2                                   <cue ref="clean up" comment="Через 10 секунд корабль-
                                    соперник прыгнет в другой сектор и там  будет
                                    уничтожен"/>
3                                </cues>
4                             </cue> <-этот тег закрывает Эпизод "coward"
```

Как мы уже говорили ранее, если Эпизод содержит атрибут ref="", то это означает фактически *"вызов"* библиотечного Эпизода с таким именем, в данном случае это снова Эпизод "clean up". Комментарий к Эпизоду призван помочь понять, что же должно произойти при его выполнении.

Вы наверное обратили внимание, что в приведенном листинге присутсвует лишняя закрывающая скобка </cue>. Это означает, что Эпизод "coward" завершен, и следует приступить к описанию следующего Эпизода, который находится на одном уровне вложенности с Эпизодом "coward", т.е. является вложенным для Эпизода "closetoplayer".

```
                              <cue name="opponentready" comment="по идее, нам не нужен
                              этот Эпизод, но учитывая особенности  действующего Flight
                              AI нам придется его написать">
                                 <condition>
                                    <check_all>
1                                      <cue_is_complete cue="closetoplayer"/>
2                                      <object_position object="createopponent.opponent"
                                       max="200m">
                                          <position object="parent.marker"/>
                                       </object_position>
                                    </check_all>
                                 </condition>
                                 <action>
3                                   <set_command object="createopponent.opponent"
                                    command="none"/>
                                 </action>
```

```
                        </cue>
```

В секции `<condition>` этого Эпизода мы проверяем следующее:

1: То что родительский, для данного, Эпизод `"closetoplayer"` выполнен/завершен (ведь несмотря не то, что текущий Эпизод не следует сразу за родительским, он все равно вложен в него) и ...

2: Что объект `"opponent"` находится на расстоянии не более 200m от объекта `"marker"`. Обратите внимание, что обращение к этому объекту происходит с использованием префикса *"parent."*, что означает, что мы обращаемся к объекту `"marker"` объявленному в родительском Эпизоде, в данном случае - в Эпизоде `"closetoplayer"`, т.е. фактически мы обращаемся к объекту `"closetoplayer.marker"`. Оба варианта равнозначны. Итак, при соблюдении обоих условий Эпизод будет активирован.

3: В блоке Действий мы даем команду объекту `"opponent"` остановиться и, прекратив всякую деятельность, оставаться на месте, для чего использовали команду *"none"*.

Следующий Эпизод, который мы рассмотрим, находится на том же уровне вложенности, что и предыдущий, т.е. является вложенным для Эпизода `"closetoplayer"`. В данном случае, вместо проверки соответствия только положения соперника относительно стартовой позиции `"marker"`, мы осуществляем проверку положения и соперника и Игрока относительно этой стартовой позиции.

```
                        <cue name="inposition">
                           <condition>
                              <check_all>
1                                <cue_is_complete cue="closetoplayer"/>
2                                <object_position object="createopponent.opponent"
                                 max="200m">
                                    <position object="parent.marker"/>
                                 </object_position>
                                 <object_position max="200m">
3                                   <position object="parent.marker"/>
                                 </object_position>
                              </check_all>
                           </condition>
                           <action>
                              <do_all>
4                                <incoming_message
                                 author="{object.pilot@createopponent.opponent}"
                                 text="{1081,8}" popup="1"/>
5                                <set_command object="createopponent.opponent"
                                 command="none"/>
                                 <set_relation object="createopponent.opponent">
6                                   <relation relation="enemy"
                                    relationobject="{player.ship}"/>
                                 </set_relation>
7                                <set_target object="S01M25S00prepare.finish"/>
                              </do_all>
                           </action>
```

Итак, аналогично предыдущему Эпизоду, мы проверяем:

1: Что Эпизод `"closetoplayer"` находится в статусе - завершен.

2: Мы снова проверяет позицию соперника относительно точки старта, он не должен находиться на расстоянии большем, чем 200m.

3: Мы так же проверяем позицию корабля Игрока относительно той же точки, и он так же не должен находиться дальше чем в 200 метрах от метки.

В случае, если все Условия Активации соблюдены, то выполняется секция Действий:

4: Игрок получает всплывающее сообщение от соперника, содержащее следующий текст `"Ок, мы оба на стартовой позиции, -начинаю отсчет..."`

5: Мы даем команду объекту `"opponent"` остановиться и, прекратив всякую деятельность, оставаться на месте, для чего и спользовали команду *"none"*.

6: Мы меняем отношение соперника, используя команду `<set_relation>`, на враждебное, по отношению к объекту `{player.ship}`. Т.е. выполняем то, что ранее соперник, в родительском Эпизоде, пообещал Игроку в переданном им сообщении и зменить работу своего сигнала свой/чужой.

7: И в завершении Эпизода мы устанавливаем игроку в качестве цели выбранную ранее в качестве *"финишной"* отметки, станцию.

Вас наверное удивило, то что мы проверили позицию соперника дважды, в разных Эпизодах. В первом случае, мы после проверки задали сопернику команду находиться в проверенной позиции, на расстоянии не более 200 метров от метки. Это было сделано для варианта развития событий, когда соперник прибудет в точку старта раньше Игрока, и в таком случае он должен остановиться и ждать. Во втором случае, мы обрабатываем вариант развития событий, когда они прибудут одновременно.

Сообщение, которое было отправлено игроку в Эпизоде `"inposition"`, является связующим звеном этого Эпизода со следующим, вложенным в него Эпизодом. В этом эпизоде мы реализовали предстартовый отсчет:

```
                        <cues>
                           <cue name="countdown" comment="count down from 4 to
                           1">
                              <condition>
1                                <cue_is_complete cue="inposition"/>
                              </condition>
                              <timing>
2                                <count exact="4"/>
3                                <time min="2s"/>
4                                <interval exact="1s"/>
                              </timing>
                              <action>
                                 <do_all>
5                                   <set_value name="this.counter" exact="5-
                                    {index@this}" comment="невозможно
                                    использовать вычисляемое значение в качестве
                                    атрибута текстового сообщения, следовательно
                                    его вначале нужно присвоить переменной"/>
6                                   <play_subtitles
                                    author="{object.pilot@createopponent.opponen
                                    t}" text="{value@this.counter}!"/>
7                                   <play_sound soundid="924"/>
                                 </do_all>
                              </action>
                           </cue>
```

Несмотря на то что, возможно он кажется несколько непростым, на самом деле этот Эпизод достаточно прост и прямолинеен.

1: Условия Активации достаточно просты, мы просто проверяем, что родительский Эпизод `"inposition"` был выполнен успешно.

В случае соблюдения Условий Активации ход переходит к секции Хронометража - `<timing>`. И именно в этом Эпизоде мы с Вами впервые встречаемся с секцией `<timing>`, содержащей подсекции `<count>`, `<time>` и `<interval>`. Давайте рассмотрим каждую из них по-отдельности:

2: `<count exact="4"/>` - означает, что действия, описанный в секции `<action>` будут выполнены ровно четыре раза.

3: `<time min="2s"/>` - означает, как и в ранее рассмотренных Эпизодах, что выполнение секции Действий будет запущено (а точнее первой итерации), как минимум, через 2 секунды после соблюдения Условий Активации.

4: `<interval exact="1s"/>` - означает, что интервал между каждым выполнение секции Действий, т.е. между каждой итерацией, будет составлять ровно одну секунду.

Теперь мы можем перейти к рассмотрению секции Действий, не забывая о том, что ее выполнение будет повторяться с интервалом в одну секунду. Особое внимание следует уделить команде `<set_value>`. И снова, для того, чтобы лучше разобраться в происходящем, мы рассмотрим каждое действие индивидуально.

5: Здесь мы присваиваем значение переменной `"this.counter"` равным 5 минус `{index@this}`. Что означает, что каждый раз, когда секция Действий будет выполняться (на каждой итерации), в нашем случае, согласно значению `<count>`, четыре раза, `{index@this}` примет значение номера итерации, выполняемой в данный конкретный момент времени. Следовательно в последовательности итераций значение `"this.counter"` будет последовательно уменьшаться с 4 до 3, затем до 2, и в завершении станет равным 1 (т.е. 5-1, 5-2, 5-3 и 5-4). Как указано в комментарии, мы не можем использовать вычисляемое значение в качестве текстового аргумента, следовательно мы вначале должны присвоить его переменной, и уже ее использовать по назначению.

6: Здесь мы осуществляем отображение субтитров на экране, отображая значение переменной `"this.counter"`, вычисленное ранее. В последовательности итераций выполнения секции Действий на экран будут выведены значения “4”, “3”, “2” и “1”.

7: В дополнении к отображению заданного числа в субтитрах, в каждую итерацию будет проигран звук, имеющий идентификатор *ID "924"* (*Внимание*). Таким образом, Игрок получит не только визуальный, но и звуковой предстартовый отсчет.

Итак мы рассмотрели Эпизод, осуществляющий предстартовый отсчет, целиком. Он не имеет вложенных Эпизодов

. Теперь перейдем к следующему Эпизоду, который так же вложен в Эпизод `"inposition"`. В нем мы проверим соответствие позиции Игрока стартовой позиции, и если Игрок попытался осуществить фальш-старт, то гонки будут прерваны.

```
                        <cue name="headstart">
                           <condition>
                              <check_all>
1                                <cue_timer cue="inposition" min="3s"
                                 max="6s"/>
2                                <object_position min="220m">
                                    <position object="closetoplayer.marker"/>
                                 </object_position>
                              </check_all>
                           </condition>
                           <action>
                              <do_all>
3                                <incoming_message
                                 author="{object.pilot@createopponent.opponen
                                 t}" text="{1081,16}" popup="1"/>
4                                <destroy_object
                                 object="closetoplayer.marker"/>
5                                <cancel_cue cue="countdown"/>
6                                <cancel_cue cue="coward"/>
7                                <cancel_cue cue="go"/>
                              </do_all>
                           </action>
```

1: Здесь мы встречаемся с новым для нас Условием Активации - `<cue_timer>` , которое проверяет время, в течении которого выполняется заданный ему в качестве параметра Эпизод. В нашем случае, мы проверяем, что Эпизод `"inposition"` уже выполняется в течении как минимум 3, но не более 6 секунд.

2: Далее мы проверяем, находится ли корабль Игрока на расстоянии не меньшем чем 220 м. от метки точки старта. Следует помнить, что если не задан объект в команде `<object_position>`, это означает, что она применяется к текущему кораблю Игрока.

Итак, если оба условия соблюдены, это означает, что Игрок попытался стартовать раньше соперника.

3: При попытке обманом выиграть деньги соперника, Игрок получит сообщение: `"Ты позорный читер! Я заметил, что ты пытался обмануть меня, осуществив фальш-старт. Пари разорвано!"`

4: Мы уничтожаем метку старта.

5-7: Отменяем/прерываем Эпизоды `"countdown"`, `"coward"` и `"go"`, а так же все их вложенные. Таким образом выполнение данной Миссии будет полностью прекращено.

И в завершении данного Эпизода (`"headstart"`), мы в очередной раз обращаемся к библиотечному Эпизоду `"clean up"`, в котором полностью удаляем все, что было создано или изменено в течении выполнения Миссии.

```
                        <cues>
                           <cue ref="clean up" comment="Через 10 секунд
                           корабль-соперник прыгнет в другой сектор и там
                           будет уничтожен">
                              <params>
1                                <param name="time" value="10s"/>
                              </params>
                           </cue>
                        </cues>
                     </cue> <-закрывающий тег Эпизода "headstart".
```

Как и ранее, мы используем одно и то же значение параметра [**1**]. Что же происходит при вызове этого библиотечного Эпизода, мы рассмотрим .

Итак, теперь мы рассмотрим, как бесстрашный пилот примет участие в этой гонке.

```
                     <cue name="go">
                        <condition>
1                          <cue_is_complete cue="inposition"/>
                        </condition>
                        <timing>
2                          <time min="7s"/>
                        </timing>
                        <action>
                           <do_all>
3                             <play_subtitles
                              author="{object.pilot@createopponent.opponen
                              t}" text="{1081,10}"/>
4                             <play_sound soundid="923"/>
5                             <set_command
                              object="createopponent.opponent"
                              command="dock"
                              commandobject="S01M25S00prepare.finish"/>
6                             <destroy_object
                              object="closetoplayer.marker"/>
                           </do_all>
                        </action>
```

1: Действие будет активировано только в том случае, когда Эпизод `"inposition"` выполнен успешно. Это очень логично, так как соперники должны находиться в равных условиях, перед началом гонки.

2: Секция Действий фактически будет активирована минимум через 7 секунд после соблюдения Условий .

3: Первым элементом секции Действия будет отображение субтитров от имени соперника с простой и понятной надписью - `"Поехали!(GO!)"`.

4: Снова прозвучит сигнал *"внимание"* (*ID 923*), в данном случае в роли выстрела из стартового пистолета.

5: В тоже самое время, когда будут отображены субтитры и прозвучит сигнал, корабль соперник получит команду на стыковку со станцией, которая нами ранее была определена в качестве *"финишной"* позиции.

6: Ну и для соблюдения чистоты и уборки после себя, мы удаляем метку старта, так как она нам больше не нужна.

Следующий Эпизод является вложенным в Эпизод "go".

```
                                    <cues>
                                       <cue name="annoyme">
                                          <condition>
1                                            <cue_is_complete cue="go"/>
                                          </condition>
                                          <timing>
2                                            <count min="3" max="6"/>
3                                            <time min="30s" max="60s"/>
                                          </timing>
                                          <action>
                                             <do_all>
4                                               <set_value name="this.txtid" min="21"
                                                max="40"/>
5                                               <incoming_message popup="1"
                                                temporary="1"
                                                author="{object.pilot@createopponent.o
                                                pponent}"
                                                text="{1081,{value@this.txtid}}"/>
                                             </do_all>
                                          </action>
6                                      </cue>
```

1: Условие Активации очень простое - полное успешное завершение родительского Эпизода `"go"`.

2: В секции Хронометража мы задаем периодичность выпол нения секции Действий в диапазоне от 3 до 6 раз. Конкретное значение числа итераций будет произвольно выбрано в этом диапазоне.

3: Так же первое выполнение секции Действий будет осуществлено через произвольное число секунд, лежащее в диапазоне между 30 и 60.

4: В этой строке мы задаем значение переменной `"this.txtid"`, это снова будет случайное значение, в данном случае лежащее в диапазоне между 21 и 40. Т.е. в каждой итерации в этой строке переменной будет присваиваться новое случайное значение из указанного диапазона.

5: В результате отработки параметров секции `<timing>` и значения `"this.textid"` будут посылаемые Игроку, через 30-60 секунд после старта гонки, всплывающие сообщения, с произвольно выбранным в каждой от трех до шести итераций текстовым со общением, имеющим индекс в диапазоне 21-40 в текстовой странице 1081. Эти сообщение имеют атрибут *"temporary"* равный 1, т.е. это *"временные"* сообщения, которые не будут сохранены в журнале сообщений Игрока. Обратите внимание, что в качестве значения атрибу та `text=""` используется переменная вычисленная в **4**.

Итак, этот Эпизод завершен, но мы не закончили рассмотрение вложенных в Эпизод `"go"` Эпизодов.

```
                                    <cue name="winner">
                                       <condition>
                                          <check_all>
1                                            <object_is_docked
                                             dockobject="S01M25S00prepare.finish"/>
2                                            <object_is_docked
                                             dockobject="S01M25S00prepare.finish"
                                             object="createopponent.opponent"
                                             negate="1"/>
                                          </check_all>
                                       </condition>
                                       <action>
                                          <do_all>
3                                            <cancel_cue cue="loser"/>
4                                            <cancel_cue cue="coward"/>
5                                            <cancel_cue cue="annoyme"/>
```

```
6                                                    <play_sound soundid="1007"/>
7                                                    <incoming_message
                                                     author="{object.pilot@createopponent.o
                                                     pponent}" text="{1081,3}" popup="1"/>
8                                                    <set_relation
                                                     object="createopponent.opponent">
                                                        <relation relation="friend"
                                                        relationobject="{player.ship}"/>
                                                     </set_relation>
9                                                    <reward_player>
                                                        <money
                                                        exact="{value@S01M25S00prepare.rewa
                                                        rd}"/>
                                                     </reward_player>
                                                  </do_all>
                                               </action>
```

1, 2 Давайте рассмотрим эти два Условия Активации в комплексе. Эти условия описывают ситуацию, когда корабль игрока уже пристыкован к *"финишной"* станции, в то время, как корабль соперника - еще не состыковался с ней. И в этом случае данный Эпизод будет активирован. Атрибут `negate="1"` означает, что мы проверяем наличие отрицательного результата условия.
Таким образом, когда оба условия соблюден ы, нам необходимо выполнить соответствующие действия.

3-5: Эпизоды `"loser"`, `"coward"` и `"annoyme"` и вложенные в них будут прерваны/отменены.

6: Будет проигран звук *"1007"* (*(unused) DING DA DONG station announcement variation 2*)

7: Игрок получит сообщение от соперника, в котором говориться "`Говорит Аргонский пилот {object.pilot@createopponent.opponent}. Ты победил! Перевожу награду в суммен {value@S01M25S00prepare.reward} кредитов на твой счет. Хорошенько отпразднуй победу в местном баре!`"

8: Следующим шагом мы восстанавливает отношение соперника к Игроку, как Вы наверное помните, мы устанавливали ему ранее статус `"enemy"` отношению к Игроку. Теперь же все возвращается на свои места и статус снова становится дружественным .

9: … и само собой, мы выдаем Игро ку обещанную награду. Сумма награды была задана в переменной `"this.reward"` объявленной в Эпизоде *S01M25S00prepare*.

Конечно, все эти действия будут выполнены только в случае победы Игрока. Если же Игрок проиграет в этой гонке, то в соответствии с условиями , которые помогут нам определить его поражение, нам нужно будет выполнить действия, соответствующие сложившейся ситуации. И мы рассмотрим их в следующем Эпизоде.

Однако нам опять нужно подчистить за собой, вызвав библиотечный Эпизод `"clean up"`.

```
                                               <cues>
                                                  <cue ref="clean up" comment="Через 5
                                                  минут корабль-соперник прыгнет в другой
                                                  сектор и там  будет уничтожен">
                                                     <params>
                                                        <param name="time" value="5m"/>
                                                     </params>
                                                  </cue>
                                               </cues>
                                            </cue> <!--закрывающий тег Эпизода „winner”.-->
```

Как и во всех ранее рассмотренных вызовах библиотечного Эпизода `"clean up"`, в нем будут очищены/удалены все объекты, созданные во время выполнения Миссии, в данном случае - через 5.

Настало время рассмотреть Эпизод "loser":

```
                                            <cue name="loser">
```

```
                                    <condition>
                                       <check_all>
1                                         <object_is_docked
                                          dockobject="S01M25S00prepare.finish"
                                          negate="1"/>
                                          <object_is_docked
                                          dockobject="S01M25S00prepare.finish"
                                          object="createopponent.opponent"/>
2                                      </check_all>
                                    </condition>
                                    <action>
                                       <do_all>
3                                         <cancel_cue cue="winner"/>
4                                         <cancel_cue cue="coward"/>
5                                         <cancel_cue cue="annoyme"/>
6                                         <play_sound soundid="1007"/>
7                                         <incoming_message
                                          author="{object.pilot@createopponent.o
                                          pponent}" text="{1081,4}" popup="1"/>
8                                         <set_relation
                                          object="createopponent.opponent">
9                                            <relation relation="friend"
                                             relationobject="{player.ship}"/>
                                          </set_relation>
                                          <reward_player>
10                                           <money exact="-
                                             {value@S01M25S00prepare.reward}"/>
                                          </reward_player>
                                       </do_all>
                                    </action>
                                    <cues>
11                                     <cue ref="clean up" comment="Через 5
                                       минут корабль-соперник прыгнет в другой
                                       сектор и там  будет уничтожен">
                                          <params>
                                             <param name="time" value="5m"/>
                                          </params>
                                       </cue>
                                    </cues>
                                 </cue> <-закрывающий тег Эпизода 'loser'.
```

В первую очередь хотелось бы отметить, что данный Эпизод в общем не сильно отличается от Эпизода `"winner"`. К тому же он также завершается вызовом библиотечного Эпизода `"clean up"` [**11**], с использованием такого же значения параметра.

1&2: Условия Активации очень похожи на те, что использованы в Эпизоде `"winner"`, с точностью до наоборот, т.е. мы проверяем выполнение условия, когда соперник пристыкован к станции, а Игрок - нет. Если данные условия выполнены - то Эпизод активируется.

3-6: Первые строки секции Действий также подобны Эпизоду `"winner"`, однако в данном случае, кроме Эпизодов `"coward"` и `"annoyme"` прерывается/отменяется выполнение Эпизода `"winner"`(и вложенных в них Эпизодов). Также звучит сигнал, аналогичный использованому в Эпизоде `"winner"`.

7: И снова, как и в Эпизоде `"winner"` Игроку отсылается сообщение от имени соперника, однак, как Вы видите, совершенно с противоположным по смыслу содержанием:

`"Говорит Аргонский пилот {object.pilot@createopponent.opponent}. Ты был близок к победе, но выиграл Я. Я лучший, я самый быстрый и ловкий! Где мои, честно заработанные, деньги!? Я готов поставить тебе выпивку за мою победу."`

8&9: Как и в Эпизоде `"winner"`, мы восстанавливаем отношение оппонента к Игроку.

10: В этом подразделе, `<reward_player>`, мы, в противоположность сделанному в Эпизоде `"winner"` снимаем деньги со счета Игрока, т.е. он проиграл свою ставку. Ранее, мы убедились, что у Игрока есть деньги, с запасом как минимум в 500 кр едитов.

Теперь Вы можете глубоко вздохнуть и потрепать себя по плечу. Миссия в общем и целом завершена.

Осталось аккуратно завершить все, а так же изучить тот библиотечный Эпизод, о котором мы говорили ранее. Очень важно помнить, что библиотечные Эпизоды должны располагаться на одном, или более высоком, уровне иерархии с местом их вызова. Так же следует помнить, что понятие *"выше"* по иерархии не соответствует понятию *"выше"* по структуре файла. Таким образом, несмотря на то, что библиотечные Эпизоды обычно располагаются в самом низу файла Миссии, это не мешает им находиться на уровне иерархии равном, или более высоком, чем места их вызова (`<cue ref="blah">`).

Следовательно, чтобы убедиться, что мы поместим наш библиотечный Эпизод в надлежащее место в иерархии, мы должны закрыть все Эпизоды, ниже (и на уровне) которых есть вызов библиотечного Эпизода.

```
                              </cue>  <-закрывающий тег Эпизода ‘go’.
                           </cues>
                        </cue>  <-закрывающий тег Эпизода ‘inposition’.
                     </cues>
                  </cue>  <-закрывающий тег Эпизода ‘closetoplayer’.
               </cues>
            </cue>  <-закрывающий тег Эпизода ‘createopponent’.
```

Итак, после закрытия скобок четырех Эпизодов, мы находимся на одном уровне с Эпизодами `"reset"`, `"selectOpponent"` и `"createopponent"`. Именно на этот уровень иерархии мы хотим поместить наш библиотечный Эпизод.

Итак приступаем к завершающему аккорду нашей Миссии … последнему кирпичику кода в ней. Библиотечный Эпизод `"clean up"` находится на одном уровне иерархии с Эпизодами `"selectOpponent"` and `"createopponent"` и состоит из одного основного и двух вложенных Эпизодов.

```
1              <cue name="clean up" library="1">
                  <timing>
2                    <time min="{param@time}"/>
                  </timing>
                  <action>
                     <do_all>
3                       <find_sector name="this.jumpsec" exact="1"/>
4                       <find_gate name="this.jumpgate" nearest="1">
                           <sector sector="this.jumpsec"/>
                        </find_gate>
5                       <set_command object="createopponent.opponent"
                        command="jumpsector"  commandobject="this.jumpgate">
                           <sector sector="this.jumpsec"/>
                        </set_command>
                     </do_all>
                  </action>
```

1: Обязательным атрибутом для любого библиотечного Эпизода является конструкция вида `library="1"`. Так же мы назначаем ему имя `"clean up"` именно его мы использовали ранее во всех значениях атрибута `ref=""` при вызове этого библиотечного Эпизода.

2: В секции Хронометража мы используем конструкцию `{param@time}` которая позволяет нам при вызове этого Эпизода задавать различное время его Активации.

3: Далее мы находим произвольный сектор, прилежащий к текущему (находящийся на расстоянии в один прыжок), и заносим его в переменную `"this.jumpsec"`.

4: Так же мы ищем Врата, ведущие в этот сектор, и запоминаем их в переменной `"this.jumpgate"`.

5: В завершении секции Действий мы даем команду кораблю соперника прыгнуть в указанный сектор, через указанные врата.

```
                  <cues>
                     <cue name="destroy opponent">
                        <condition>
1                          <object_changed_sector object="createopponent.opponent" />
                        </condition>
                        <timing>
2                          <time min="1s"/>
                        </timing>
                        <action>
                           <do_all>
3                             <destroy_object object="createopponent.opponent" />
                           </do_all>
                        </action>
                     </cue>
```

1: Как только соперник покидает сектор с Игроком (меняет сектор своего местоположения) - условие соблюдено.

2: Через одну секунду после того как он покинул сектор с Игроком…..

3: … его корабль будет уничтожен.

Но это еще не все. Имя следующего Эпизода и комментарии к нему все х орошо объясняют

```
                     <cue name="restart" comment="делаем нашу Миссию снова доступной
                     для Игрока">
                        <timing>
1                          <time min="1h"/>
                        </timing>
                        <action>
2                          <reset_cue cue="S01M25S00prepare"/>
                        </action>
                     </cue>
                  </cues>
               </cue> <-закрывающий тег библиотечного Эпизода „clean up"
            </cues>
```

1: Через час после уничтожения корабля соперника (обратите внимание на отсутствие Условий Активации)…

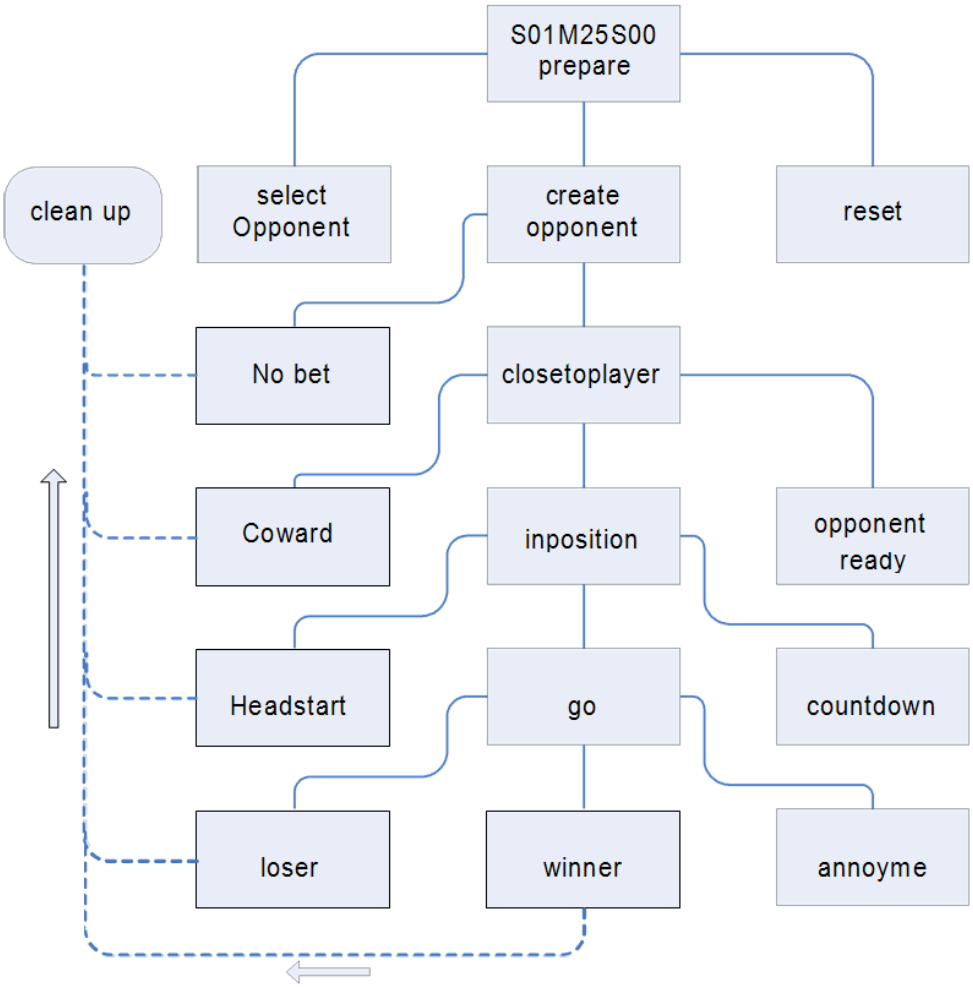
2: Эпизод верхнего уровня всей Миссии перезапускается и готов к исполнению снова.

Итак, мы достигли конца Миссии … и нам осталось только разместить закрывающие скобки для конструкций самого верхнего уровня, а именно: `</cue>`, `</cues>` и `</director>`.

```
         </cue> <-закрывающий тег Эпизода 'S01M25S00prepare'.
      </cues>
   </director>
```

Ну что ж, мы сделали это. Одну, но полнофункцио нальную миссию.

Если вы попытаетесь визуализировать структуру Миссии, то вы должны получить нечто, подобное этому рисунку:



Для своих Миссий Вы так же можете рисовать подобные диаграммы. Они должны вам помочь в понимании того, что же у Вас происходит и как отдельные Эпизоды взаимодействуют между собой. Визуализация структуры Миссии в виде диаграммы очень помогает при разработке.

Руководство по написанию миссий, доступных в виде объявлений на станциях (BBS миссий)

Одновременно с изучением правил создания Миссий, события которых происходят в открытом космосе, нам стоит разобраться, как написать Миссию, которая будет доступна Игроку в виде сообщения на станции, т.е. BBS Миссию. Это т тип Миссий заслуживает отдельного рассмотрения, потому что при их написании стоит учитывать некоторые их структурные отличия. Шаблон BBS Миссии поставляется вместе с документацией к Mission Director-у, и находится в папке примеров и образцов (*samples)*, которая является вложенной в папку *director*. В нем уже отображена вся необходимая структура файла Миссий, которая обеспечит корректную работу BBS Миссии. Имеет смысл попытаться самостоятельно изучить данный шаблон и поэкспериментировать с ним, прежде чем пи сать собственную BBS Миссию.

Как и в предыдущем примере, мы начнем с заголовка и секции документации. Эти части файла Миссий не зависят от типа Миссии:

```
<?xml version="1.0" encoding="utf-8" ?>
<?xml-stylesheet href="director.xsl" type="text/xsl" ?>
<director name="test" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="director.xsd">
   <documentation>
      <author name="Fred Bloggs" alias="Froggs" contact="froggs@egosoft.com"/>
      <content name="Lottery BBS" description="BBS-based lottery mission"
      reference="B01M47S00_XXXT"/>
      <version date="29 May 2007" number="1.0" status="Beta Test"/>
   </documentation>
```

Опять же, как и в предыдущем примере, необходимо обеспечить уникальность именования Эпизода, во избежание возможных конфликтов с Миссиями написанными другими авторами. Это означает, что наименования Эпизодов содержат уникальную приставку, связанную со ссылкой на Миссиию, к примеру: `B01M47S00MainCue`.

```
   <cues>
1     <cue name="B01M47S00LotteryMainCue" comment="Это Эпизод верхнего уровня,
      стартовая позиция всей нашей Миссии">
         <condition>
2           <check_age value="{player.age}" min="5s" comment="Условия,
            определяющие, как, где и в каком случае объявление с предложением ББС
            Миссии будет отображено на доске объявлений"/>
         </condition>
         <action>
3           <do_all>
4              <add_bbs_quest name="B01M47S00Lottery" priority="500" max="1"
               comment="Priority= задает частоту отображения Миссии на доске
               объявлений, диапазон значений 0-100. Max= максимально экземпляров
               объявлений с предложением Миссии на доске объявления одной
               станции"/>
            </do_all>
         </action>
```

1: Разрешите представить Вашему вниманию Эпизод верхнего уровня в данной Миссии, наименование данного Эпизода содержит описанную нами приставку, ссылающуюся на миссию, и соответствующий комментарий. Мы использовали шаблон, и используя комментарии, попытались пояснить, за что отвечает каждый элемент в данной Миссии.

2: Условия Активации данного Эпизода чисто *"отладочные"*; это означает, что с целью тестирования работоспособности Миссии, Эпизод активируется через 5 секунд после старта новой игры. Естественно, Вы всегда сможете использовать весь спектр возможных Условий Активации при написании ББС Миссий.

3: Так как фактически у нас только одна команда в секции Действий, то тег `<do_all>` использован только с целью предоставления возможности в дальнейшем, в случае потребности, добавить еще какие-либо команды в данную секцию.

4: Это основополагающая команда для всех ББС Миссий. `<add_bbs_quest>`, как следует из ее названия, она добавляет ББС Миссию в список Миссий, которые обрабатываются движком ББС Миссий. Аргумент `priority`(приоритет) определяет, как часто движок ББС Миссий будет предлагать указанную Миссию Игроку на доске объявлений станции. Стандартные значения приоритета лежат в диапазоне от 0 до 100. Однако в отладочных целях мы использовали число 500, для гарантированного появления предложения нашей Миссии на доске объявлений. Аргумент `max` задает максимальное количество одновременных предложений одной миссии на одной и той же станции. Стандартно он равен единице..

Следующий Эпизод является вложенным в Эпизод верхнего уровня, и в нем мы задаем отображение нашей Миссии на доске объявлений станции, после добавления Миссии к списку доступных:

```
            <cues>
1              <cue name="B01M47S00Lottery Mission Offer" instantiate="static"
               comment="Отображает сообщение на доске объявлений с предложением данной
               Миссии">
                  <condition>
2                    <bbs_quest_evaluated quest="B01M47S00Lottery" comment="Активирует
                     Миссиию согласно заданным значениям приоритета и количества
                     экземпляров"/>
                  </condition>
                  <action>
3                    <offer_bbs_quest quest="B01M47S00Lottery" author="Лотерея
                     Spamelot" text="[center]Купите лотерейный билет и выиграйте не
                     менее 1,000,000 Cr?\n\nПо цене всего в 1,000 Cr!\n\n[select
                     value='yes']Да, да, да![/select]\n[select value='no']Забудьте обо
                     мне![/select][/center]"/>
                  </action>
```

1: Как мы уже говорили, этот Эпизод вложен в Эпизод верхнего уровня Мисии. Надеемся, Вы помните, что тег `<cues>`, следующий сразу за закрывающим тегом `</action>` означает начало секции вложенных Эпизодов. Данный эпизод задает отображение БСС Миссии на доске объявлений станции. Атрибут `instantiate="static"` означает, что будет создана самостоятельна копия этого Эпизода и в дальнейшем Игрок будет "общаться" с этой копией. Другие копии будут доступны на других станциях. Более детальное описание механизма создания экземпляров Миссий и Эпизодов представлено в Приложении 1: Экземпляры.

2: В качестве Условия Активации мы ставим соблюдение приоритета и количества экземпляров отображения движком игры. Т.е. если движок ББС Миссий посчитает, что согласно заданным условиям объявление должно быть отображено, то и данный Эпизод будет активирован.

3: Далее, сам текст объявления с предложением Миссии. Здесь мы задали автора и текст самого объявления (само собой, при реализации полноценных миссий лучше использовать ссылки на текст в текстовых файлах). Итак, мы за 1,000 Cr предлагаем выиграть не меньше миллиона…

Следующий Эпизод, слова вложен в рассмотренный ранее, он определяет поведение Миссии в том случае, если Игрок ответил "Да, да, да!" и принял предложенную Миссию.

```
               <cues>
                  <cue name="B01M47S00 Mission Accepted" comment="Если Игрок принял
                  предложенную Миссию">
                     <condition>
                        <check_all>
1                          <bbs_quest_selected quest="B01M47S00Lottery"
```

```
                        answer="yes"/>
                  </check_all>
               </condition>
               <action>
                  <do_all>
2                    <do_choose>
3                       <do_when value="{player.money}" min="1001">
4                          <accept_bbs_quest quest="B01M47S00Lottery"/>
5                          <reward_player>
                              <money exact="-1000"/>
                           </reward_player>
                        </do_when>
6                       <do_otherwise>
7                          <incoming_message author="Лотерея Spamelot"
                           text="У Вас недостаточно денег для покупки
                           лотерейного билета. Всего хорошего."/>
8                          <cancel_cue cue="lucky dip"/>
                        </do_otherwise>
                     </do_choose>
                  </do_all>
               </action>
```

1: Если предложенная Миссия выбрана Игроком, т.е. он ответил "Да", то текущий Эпизод будет активирован.

2: На данном примере Вы можете прекрасно разобраться с принципом действия команды `<do_choose>`, в данном случае она содержит две подсекции, `<do_when>` и `<do_otherwise>`, в данном случае вся команда `<do_choose>` может быть описана как *"Если x=y то выполняем секцию a, иначе выполняем секцию b"* .

3: Итак, *"если у Игрока есть как минимум 1001Cr, то …"*

4: Этим мы даем команду движку ББС-миссий считать данную Миссию активной.

5: `<reward_player>` используется с отрицательным значением параметра денежного вознаграждения, т.к. это самый простой способ снять деньги со счета Игрока.

6: Теперь *"иначе, если у Игрока денег меньше, чем 1001Cr то …"*

7: … Игрок получает сообщение от компании, проводящей Лотерею , о том, что он пытался купить лотерейный билет, не имея достаточно средств для этого.

8: Для полного завершения Миссии, в случае, если у Игрока недостаточно средств для покупки билета, соответствующие Эпизоды прерываются/отменяются.

```
               <cues>
                  <cue name="lucky dip">
                     <condition>
1                       <cue_is_complete cue="parent"/>
                     </condition>
                     <action>
                        <do_all>
2                          <set_value min="1" max="7" name="this.rand1"/>
3                          <set_value min="8" max="17" name="this.rand2"/>
4                          <set_value min="18" max="25" name="this.rand3"/>
5                          <set_value min="26" max="33" name="this.rand4"/>
6                          <set_value min="34" max="42" name="this.rand5"/>
7                          <set_value min="43" max="49" name="this.rand6"/>
8                          <incoming_message author="Лотерея Spamelot"
                           text="Ваши счастливые номера: {value@this.rand1}
                           {value@this.rand2} {value@this.rand3}
                           {value@this.rand4} {value@this.rand5}
                           {value@this.rand6}\n\nУдачи!"/>
                        </do_all>
                     </action>
```

1: Данный Эпизод будет активирован только после полной отработки родительского Эпизода.

2-7: Здесь мы задаем значения 7 переменных, путем выбора случайного числа в заданных диапазонах.

8: И затем сообщаем Игроку его номера. Как были сгенерированы эти номера, не так уж и важно, так как выигрыш ими фактически не определяется.

Этот Эпизод вложен в описанный ранее `"lucky dip"`.

```
                        <cues>
                           <cue name="draw" comment="по прохождении
                           определенного промежутка времени">
                              <timing>
1                                <time min="20s" max="25s"/>
2                                <!--20-25s задано только для удобства отладки--
                                 >
3                                <!--<time min="10h" max="12h"/>-->
                              </timing>
                              <action>
                                 <do_all>
4                                   <set_value name="this.prize" min="1000000"
                                    max="1500000" comment="Сумма выигрыша в
                                    диапазоне между 1 and 1.5 миллионами Cr"/>
5                                   <set_value name="this.lotterynum" min="1"
                                    max="100000" comment="the odds of winning
                                    are 1 in 100000"/>
                                    <do_choose>
6                                      <do_when value="{value@this.lotterynum}"
                                       max="1">
7                                         <incoming_message author="Лотерея
                                          Spamelot" text="Поздравляем! Вы
                                          выиграли джек-
                                          пот!\n\n{value@this.prize} Деньги
                                          будут тут же перечислены на ваш
                                          счет."/>
8                                         <reward_player>
                                             <money exact="{value@this.prize}"/>
                                          </reward_player>
                                       </do_when>
9                                      <do_otherwise>
10                                        <incoming_message author="Spamelot
                                          Lottery" text="К сожалению, Вы ничег
                                          оне выиграли. Попробуйте еще раз."/>
                                       </do_otherwise>
                                    </do_choose>
                                 </do_all>
                              </action>
                           </cue>
                        </cues>
```

1-3: Как Вы можете видеть, на период отладки, мы задаем очень краткий период ожидания, после активации. Когда же данная миссия будет считаться рабочей, то в данной секции Хронометража будут проставлены несколько иные значения задержки, от 10 до 12 часов после покупки лотерейного билета.

4: Здесь мы задаем значение выигрыша в данном туре Лотереи, и заносим его в переменную `{this.prize}`. Это случайное число, лежащее в диапазоне от 1 до 1.5 миллионов.

5: Теперь определим вероятность выпадения Игроку счастливого билета - для этого зададим в переменную `"this.lotterynum"` случайное значение в диапазоне от 1 до 100,000. Таким образом - шанс выигрыша для Игрока - один из ста тысяч..

6: Далее, используя `<do_choose>`, мы, фактически, применяем следующее условие „если значение `"this.lotterynum"` равно 1 (т.е. было выбрано случайно из 100,000 вариантов), то должны быть выполнены вложенные действия".

7: И вот этот шанс, один из ста тысяч выпал, и мы отправляем Игроку сообщение с поздравлением, и обещаем тут же перевести выигрыш на его счет, в сумме `{value@this.prize}`, которая была задана нами ранее

8: Вот и пришло время добавить выигрыш на личный счет Игрока. Для этого мы передаем положительное значение выигрыша (оглашенное строкой выше), как параметр в команду `<reward_player>`.

9&**10**: Если же мы ранее получили значение, лежащее в диапазоне между 2 и 100,000 - то предлагаем игроку сделать еще одну попытку (в данном случае - используется тег `"otherwise"`).

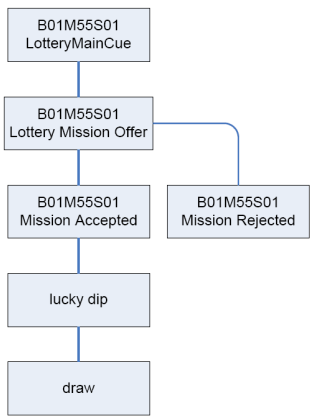
```
                        </cue> <- закрывающий тег Эпизода „lucky dip”.
                     </cues>
                  </cue> <-закрывающий тег Эпизода „B01M47S00 Mission Accepted”.
1                 <cue name="B01M47S00 Mission Rejected" comment="Если Игрок
                  отказался принять участие в нашей Лотерее">
                     <condition>
2                       <bbs_quest_selected quest="B01M47S00Lottery" answer="no" />
                     </condition>
                     <action>
                        <do_all>
3                          <incoming_message author="Лотерея Spamelot" text="Очень
                           жаль! А ведь вам могло повезти." />
                        </do_all>
                     </action>
                  </cue>
               </cues>
            </cue> <-закрывающий тег Эпизода „B01M47S00Lottery Mission Offer”.
         </cues>
      </cue> <-закрывающий тег Эпизода „B01M47S00LotteryMainCue”.
   </cues>
</director>
```

1: Если Игрок отверг наше предложение на Доске Объявлений, то управление будет передано этому Эпизоду. Заметьте, что он находится на одном уровне иерархии с Эпизодом `"B01M47S00 Mission Accepted"`.

2: Аналогично тому, как активировался Эпизод, связанный с принятием предложенной Миссии, так же активируется и Эпизод, отвечающий за отказ от миссии, только обрабатывается другое значение параметра.

3: Здесь мы просто отправляем Игроку сообщение о том, что мы сож алеем об его отказе.

Вот собственно и вся Миссия ББС. Достаточно простая, но содержащая все основные особенности такого рода Миссий.



Переменные - списки выбора (A variable worth knowing about)

В ваше распоряжение предоставлен довольно интересный тип переменных - списки объектов, отформатированные для осуществления выбора одного из объектов игроком. Такие переменные имеют следующий вид - `{group.object.select@group}`. К примеру, если в вашей миссии требуется, чтобы игрок выбрал одну станцию из некоего списка, вы можете отправить ему сообщение, содержащее такой список, совершенно не задумываясь, как оно должно быть отформатировано, сколько там элементов и т.д.

Первое и фактически единственное, что вы должны сделать - это задать группу объектов (собственно - основу формирования списка).

Например, сформируем группу станций из перечня станций в секторе, в котором находится игрок:

```
      <action>
         <do_all>
1           <find_station  class="station"  race="{player.sector.race.name}"  max="5"
            group="Foundstations" multiple="1">
               <sector sector="{player.sector}"/>
            </find_station>
         </do_all>
      </action>
```

1: Используя `find_station` мы попытаемся найти не более 5 станций, которыми владеет раса, владелец сектора, в котором они расположены, и все эти станции должны находится в том же секторе, что и игрок. Группа найденных станций будет помещена в переменную `"Foundstations"`. Так же нами было задано значение параметра `"multiple"` равным 1, иначе мы бы получили после осуществления поиска в результате максимум одну станцию.

Теперь, мы можем сформировать сообщение, в котором обратимся к игроку с запросом - выбрать одну из предложенных станций, для формирования списка станций мы используем `{group.object.select@group}`.

```
      <action>
         <do_all>
1           <do_if min="1" value="{group.object.count@Foundstations}">
2              <offer_bbs_quest                                    quest="Bargains"
               author="{random.pilot@{player.sector.race.name}}"   text="Вот   список
               станций, на которых Вы можете купить требуемый товар. Сделайте свой
               выбор.\n\n [center]{group.object.select@Foundstations}[/center]"/>
            </do_if>
         </do_all>
      </action>
```

1: Вначале мы проверяем, что группе найденных станций состоит как минимум из одной станции.

2: Если ББС квест уже имеет статус - предложен к принятию, то формируем от имени произвольного персонажа, принадлежащего к расе-владельцу текущего сектора. В сообщении мы предлагаем игроку выбрать одну из пяти станций для осуществления покупки некоего товара.

```
      <cue name="BBS Mission Accepted">
         <condition>
1           <bbs_quest_selected quest="Bargains"/>
         </condition>
```

1: Обычно условие `<bbs_quest_selected>` требует наличия двух параметров - наименования квеста, и переменной, в которую помещен ответ на запрос, сформированный в сообщении квеста. В данном случае мы не используем этот параметр, и потому результат ответа на запрос квеста просто остается в памяти.

Для того, чтобы сделать доступным для дальнейшего использования результат выбора пользователя, мы должны сохранить его в виде переменной, содержащей выбранный.

```
        <action>
1          <set_object name="this.selection" value="{quest.answer@Bargains}"/>
        </action>
```

1: Таким образом, мы присваеваем переменной „this.selection" объект, выбранный в результате обработки запроса квеста. Обратите внимание, что наименование квеста и результат обработки запроса разделены символом @.
Теперь выбранный объект(`"this.selection"`) может быть использован в любом условии или действии, в процессе работы миссии.

Данный механизм формирования и обработки запросов, содержащий список объектов может быть использован не только в запросах, сформированных ББС квестом, но и в обычных запросах-сообщениях. После осуществления выбора, результат запроса должен быть сохранен в нужную вам переменную. В принципе все аналогично предыдущему описанию, но есть несколько неболь ших отличий.

```
        <action>
1          <set_object name="this.selection" value="{question.answer@missiles}"/>
        </action>
```

1: Мы используем имя запроса (отдельного сообщения), а не имя ББС квеста.

Это значит, что вы должны были ранее использовать конструкцию `<ask_question>`, задав ей имя, использованное в примере выше. Например, так:

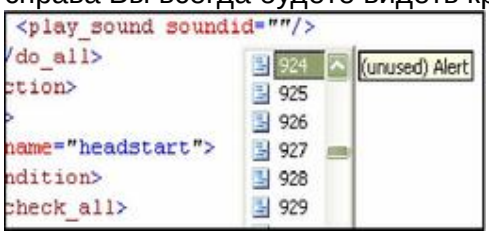
```
     <action>
        <do_all>
           <find_station class="mine" max="10" group="mines" multiple="1">
              <sector sector="{player.sector}"/>
           </find_station>
           <ask_question name="select" author="Security" text="В секторе обнаружено
           около 10 мин, и они должны быть разминированы. Пожалуйста сделайте выбор
           мины, которую Вы уничтожите первой, с целью восстановления нормального
           движеня кораблей в секторе. \n\n[center] {group.object.select@mines}
           [/center]"/>
        </do_all>
     </action>
     <cues>
        <cue name="identification">
           <condition>
              <question_answered question="select"/>
           </condition>
           <action>
              <do_all>
                 <set_object name="this.selection" value="{question.answer@select}"/>
                 <set_target object="this.selection"/>
              </do_all>
           </action>
        </cue>
```

Использование звуков (Incorporating In-Game sounds)

При написании миссий Вы можете проигрывать различные звуки, как сигналы, таки целые фразы (пример применения звуков Вы можете увидеть, заглянув в приведенные учебные примеры миссий) . Вот так может выглядеть проигрывание сигнала - предупреждения:

```
<play_sound soundid="924"/>
```

Использовать таких, встроенные в игру звуки, не просто, а очен ь просто. Если при написании миссии Вы выбрали тег команды "soundid", перед Вами тут же появится список возможных значений идентификаторов встроенных в игру звуков. Перемещаясь по списку, справа Вы всегда будете видеть краткое описание данного звука:



Полный список идентификаторов встроенных в игру звуков Вы всегда сможете посмотреть, открыв файл director.htm в Интернет Эксплорере.

Использование текстовых файлов (Incorporating texts)

В учебных примерах миссий Вы можете видеть множество вариантов использования различных текстовых строк. Они используются в сообщениях, направляемых игроку, в объявлениях на ББС, различных за просах, Вы можете писать абсолютно любой необходимый Вам текст. Мы бы рекомендовали Вам как можно реже применять текстовые строки, описанные напрямую в файле миссий. Более правильным будет применение идентификаторов текста из текстовых файлов.

Т.е. вместо прямого написания текста в файле миссии….

```
<incoming_message author="Shipboard Computer" text="Hello World"/>
```

Мы должны использовать ссылку на текстовую страницу и текст в ней…

```
<incoming_message author="{1323,119}" text="{1278,305)"/>
```

Тестирование Ваших миссий (Testing your missions)

Очень важно всегда проверять работоспособность и корректность поведения Вашей миссии на этапе ее разработки. Об этом необходимо заботится в течении всего процесса написания Миссий. Если Вы будете регулярно проверять, как ведет себя Миссия, в зависимости от изменения Условий Активации и Действий, Вы сможете оперативно реагировать на возникающие ошибки и изменять поведение Миссии, практически на лету.

Меню Mission Director-a

Для того, чтобы доступиться в меню Mission Director-a Вам необходимо нажать клавишу Enter на цифровой клавиатуре, для доступа в главное меню, и выбрать пункт Mission Director, в разделе меню Игрока, как показано на приведенном скриншоте.

Данное меню предоставляет Вам для использования две функциональные возможности. Первая - просмотр всех эпизодов, составляющих Вашу миссию. Вторая, и более важная и полезная, это возможность перезапуска Mission Director-a без выхода и перезапуска игры. Однако, Вам следует учитывать, что при этом не происходит перезапуска ББС-миссий и все объекты, созданные до перезапуска продолжат свое существование в игре. Если Вам необходима полная переинициализация миссии, Вам придется начать новую игру (или загрузить игру, по состоянию до начала миссии).

Как уже неоднократно упоминалось, XML файлы миссий должны находиться в подпапке director папки с игрой, для того, чтобы они были подгружены в игру при ее старте или загрузке. И потому, мы бы рекомендовали Вам очистить эту подпапку от других миссий, для упрощения процесса тестирования, и исключения влияния этих миссий на Вашу. Файлы вида *"dir*"* обязательно должны быть в наличии в этой подпапке.

Как мы описывали ранее, приступая к тестированию Вам необходимо начать новую игру (не по сценарию). Так же мы рекомендовали бы Вам прописать в миссию некие действия, производимые сразу при старте игры. Таким образом, если Вы не наблюдаете результатов нужных Вам действия, Вам необходимо выйти из игры в главное меню, и запустить ее снова, выбрав пункт "Своя игра".

Использование файла director.dmp

Этот файл, который создается в папке с игрой, предоставит Вам очень ценную и полезную информацию для тестирования и отладки Ваших миссий.

Файл дампа состоит из трех секций:

- Первая часть довольно сложна к пониманию, при открытии файла как файла XML.

- Во второй секции содержится максимально полная информация по Эпизодам и и содержимому.
- В последней секции содержится промаркированная временными метками последовательность событий и сообщений, результатов обработки Условий Активации и Действий по выполнению каждого Эпизода.

Первое, что Вам необходимо сделать, это найти все вхождения в этот файл слова „error‟. Просмотр и анализ записей, содержащих данное слово должно помочь в решении проблем с Вашими миссиями.

Если же последняя секция практически пустая - это означает, что выполнение вашей миссии было прервано, даже если вы не обнаружили никаких ошибок при анализе данного файла.

При тестировании Ваших миссий Вам естественно необходимо знать, что же происходит с вашими Эпизодами в любой период времени их исполнения, следовательно, изучая запись в файле дампа, относящуюся к интересующему Вас Эпизоду, Вы сможете отследить какие Условия Активации были соблюдены или не соблюдены, какие Действия были либо не были выполнены, и по какой причине. Если же Вы не обнаружили какой-либо Эпизод в файле, значит проблемы возникли в родительском Эпизоде. Если же в файле присутствует записи относящиеся к Эпизоду (обычно, как минимум, *"Conditions checked (first time)"*), то Вы можете быть уверены в том, что Эпизод был принят к обработке, и была проведена проверка соблюдения Условии Активации, отследите последующие появления Эпизода в файле и сможете увидеть результат проверки соблюдения Условий Активации. Если же кроме самой первой записи (см. ранее) Вы не можете обнаружить ни одной записи связанной с Эпизодом, это значит, что Условия Активации не были соблюдены, и Вам стоит проверить заданные Вами Условия Активации в самом Эпизоде.

Если же Вы в своих миссиях используете переменные (как локальные, так и глобальные), то изучая файл дампа Вы сможете увидеть, какие значения они принимали (и были ли им присвоены какие-либо значения), что должно помочь Вам найти потенциальное место преткновения, при поиске ошибок вычислений. Если же у Вас нет уверенности в том, какие значения принимают некоторые Ваши переменные, Вы всегда можете использовать, с целью отладки, отправку сообщений в журнал игрока, в которых будут отображены текущие значения выбранных переменных. Так же, очень удобно, для проведения отладки, то, что все события, как-либо относящиеся к MD обязательно будут представлены в файле дампа. Следовательно, если Вы в Вашей миссии проверяете происхождение какого-либо события, и при этом Условия Активации, заданные Вами не были соблюдены, Вы сможете отыскать интересующее Вас событие, и увидеть, происходило ли оно на самом деле, и когда, и понять, почему оно не было обработано вашим Эпизодом.

Локализация проблем

Если в Вашей миссии присутствует множество Условий Активации, и Вы не уверены по какой причине Ваша миссия не выполняется, имеет смысл применить "локализацию проблемы". Это означает, что Вы, разделяя Условия Активации и Эпизоды, должны провести их тестирование независимо друг от друга. Для этого Вам необходимо скопировать Эпизод и его Условия Активации в новый файл, дать ему другое имя, и провести тестирование и отладку этого отдельного Эпизода (при необходимости, создайте родительский Эпизод, для задания некоторых начальных значений переменных).

Глоссарий (Glossary)

- **Секция Действий** (**Action**) – это ключевая секция в Эпизоде, в ней описаны все действия, которые должны произойти в Игре, согласно Условий Активации и во время, определенное секцией Хронометража.
- **Атрибуты (Attribute)** – обеспечивает предоставление дополнительной информации касательно секции (под-секции), с которой они ассоциированы.
- **Условия Активации (Condition)** – ключевая секция Эпизода; в ней задаются условия, при соблюдении которых, либо при происхождении событий, указанных здесь же, должны быть выполнены определенные действия в Игре.
- **Эпизод (Cue)** – ключевой элемент структуры файла Mission Director-а, содержащий секции Условий Активации, Хронометража и Действий. Так же может содержать вложенные Эпизоды.
- **Выражение (Expression)** – набор математических операций, в которых используются числа и/или переменные.
- **Экземпляр (Instance)** – копия Эпизода, создаваемая при задании атрибута `instantiate` , существующая и обрабатываемая отдельно от оригинала.
- **Внутриигровой идентификатор (Internal ID)** – каждый объект в игре имеет свой внутриигровой идентификатор, присущий только ему, и его однозначно его идентифицирующий.
- **Библиотечный Эпизод (Library cue)** –Эпизод, секции которого, либо он весь целиком может быть использован, при задании атрибута `ref=""`, с именем этого Эпизода, в месте вызова. В остальных случаях библиотечные Эпизоды не обрабатываются и не разбираются (парсятся) ...
- **Искомые значения (Look-up)** – выпадающее меню с возможными значениями для данного атрибута.
- **Секция (Node)** – тег XML, обрамляющий некую последовательность других тегов.
- **Объект (Object)** – в понятиях Mission Director-а, это игровой объект, существующий в игре, либо создаваемый в процессе исполнения Миссии.
- **Объект (Object)** – в понятиях Mission Director-а, это игровой объект, существующий в игре, либо создаваемый в процессе исполнения Миссии.
- **Парсинг, Обработка (Parsing)** – обычно означает "считывание и обработка", т.е. это процесс с помощью которого игра воспринимает XML файлы Mission Director-а.
- **Квест (Quest)** – Миссия, получаемая на Доске Сообщений, в Штаб-Квартирах, в Космосе, или связаны с Сюжетом.
- **Схема (Schema)** – XML файл, который описывает структуру секций, подсекций и искомых значений.
- **Вложенный Эпизод (Sub-cue)** – Эпизод, описанный в теле другого Эпизода, на который влияют Условия Активации родительского, т.е. он никогда не будет активизирован, если не были соблюдены Условия Активации родительского Эпизода.

- **Подсекция, вложенная секция (Sub-node)** – вложенная секция, подобная атрибуту функции, однако содержащая и группирующая несколько записией с различной информацией.

- **Секция Хронометража (Timing)** – ключевая секция Эпизода, в которой задаются промежуток времени между соблюдением Условий Активации и активацией секции Действий, а так же количество итераций исполнения секции Действий, а так же интервалы между итерациями.

- **Значение (Value)** – это могут быть любые данные; строка, целое число, переменная. В игре используется множество различных значений различными способами.

- **Переменная (Variable)** – значение (текст или число), которое может быть использовано или создана в качестве заменителя переменных значений в миссиях там где нельзя использовать фиксированные значения.

Приложения (Appendicies)

В данных приложениях мы постарались предоставить Вам довольно важную информацию, которую, из-за ее объема и некоторой сложности для понимания, посчитали нужным вынести из основной части данного Руководства.

Приложение 1. Экземпляры (Instantiation)

Использование, или не использование, экземпляров Эпизодов определяет поведение Эпизода после того, как его Условия Активации соблюдены:

- если не требуется создание экземпляра Эпизода, то выполняется секция Действий Эпизода (через промежуток времени, определенный в секции Хронометража), и затем Эпизод получает статус - выполнен/завершен;
- если Эпизод описан, как требующий создание экземпляров, тогда создается КОПИЯ Эпизода (и всех вложенных в него Эпизодов), и именно в этой КОПИИ (экземпляре) запускается на выполнение секция Действий, после завершения выполнения которой именно КОПИЯ(экземпляр) получает статус выполнена/завершена. И следовательно, если Условия Активации Эпизода снова будут соблюдены в дальнейшем, то история повторится.

Эпизоды, которые отмечены как требующие создания экземпляров, должны иметь Условия Активации, которые могут быть соблюдены единожды (в крайнем случае - ограниченное число раз) или эти Условия должны включать в себя реакцию на какое-либо событие.

Экземпляры Эпизодов не следует использовать в случае, если в Условиях Активации проверяется только зависимость от времени, прошедшего с начала игры, потому что каждые несколько миллисекунд будут создаваться рабочие экземпляры таких Эпизодов, что приведет к "вылету" игры.

Наиболее логично использовать экземпляры для Эпизодов, которые должны активироваться при смене сектора, кораблем Игрока, или реагировать на принятие Игроком Миссий с Доски Объявлений.

Так как процесс, создающий экземпляр Эпизода, вместе с ним копирует и все вложенные в него Эпизоды, вам следует разумно подходить к использованию экземпляров Эпизодов, а особенно быть внимательным в написание Эпизодов, чьи предки отмечены, как требующие создания экземпляра. Так же это означает, что вам не стоит забывать о том, что оригинальный Эпизод может быть активирован сразу после того, как был создан его рабочий экземпляр. Кроме того, стоит помнить, что после активации, экземпляр Эпизоды будет помечен как полностью завершенный, и выгружен из памяти только после того, как все вложенные в него Эпизоды будут выполнены, либо отменены, на всех уровнях иерархии вложенности.

Так же стоит очень осторожно относиться к маркировке Эпизодов, как требующих создания экземпляров, еще и потому, что Вы можете получить ситуацию, когда из созданного экземпляра Эпизода верхнего уровня, будут созданы экземпляры вложенных в него Эпизодов, которые создадут свои копии иерархии вложенных Эпизодов. Что может привести, к примеру, к тому, что событие смены сектора Игроком будут обрабатывать несколько активных экземпляров(копий) одной и той же Миссии ББС (это произойдет потому, что условия данной Миссии были приняты игроком, и она находится в активном состоянии).

Приложение 2. Библиотечные эпизоды (Library Cues)

В MD существует возможность создавать повторно используемые Эпизоды, которые носят название библиотечных. К примеру, пусть у нас есть несколько разных Эпизодов, в которых нужно создать случайно выбранный корабль класса M5, и если мы используем такой библиотечный Эпизод, в котором и запишем код создания корабля, то нам не понадобиться копировать и переносить изменения из одного Эпизода в другой, помнить, где и что мы изменили, и рисковать потерей этих изменений, при переносе данных в ошибочном направлении. Таким образом, создав одиночный Эпизод, и пометив его как библиотечный, мы избавимся от указанных проблем. Кстати, следует помнить, что библиотечный эпизод никогда не будет активирован, как любой обычный, а только в случае явного его использования (вызова) в другом Эпизоде.

К примеру:

```
<cue name="mylibrarycue" library="1">
   ...
</cue>
...
<cue name="myfirstcue">
   ...
   <cues>
      <cue ref="mylibrarycue"/>
   </cues>
</cue>
...
<cue name="mysecondcue">
   ...
   <cues>
      <cue ref="mylibrarycue"/>
   </cues>
</cue>
```

В приведенном выше примере, Эпизод с именем "mylibrarycue" в процессе работы Миссии будет проигнорирован, так как он помечен как библиотечный (library="1"), однако его код будет использован, фактически он заменит ссылки на него во вложенных Эпизодах Эпизодов "myfirstcue" и "mysecondcue", и именно там, где на него описаны ссылки, он будет активирован как и любой обычный Эпизод, как если бы мы явно прописали его код там.

Иногда бывает необходимо использовать достаточно сложные, но идентичные, Условия Активации для активации нескольких разных Эпизодов, либо наоборот, используя различные Условия Активации, активировать одинаковые секции Действий. В этом случае так же можно использовать библиотечные Эпизоды, однако ссылаться на них нужно из той секции (Условия Активации, Хронометража, Действий), которую мы хотим использовать в данном конкретном случае.

К примеру:

```
<cue name="mylibrarycue" library="1">
   <action>
      ...
   </action>
</cue>
...
<cue name="myfirstcue">
   <condition>
      ...
   </condition>
   <action ref="mylibrarycue"/>
</cue>
...
```

```
<cue name="mysecondcue">
   <condition>
      ...
   </condition>
   <action ref="mylibrarycue"/>
</cue>
```

И наконец, как мы убедились, библиотечные Эпизоды очень удобны, но что делать, если нам нужно выполнить схожие действия, н о все-таки чуть-чуть отличающиеся между собой. К примеру - создать случайный корабль класса M5, и установить его принадлежность к определенной расе. Вы конечно можете сделать библиотечный Эпизод, который создаст корабль, и потом, уже в отдельном Эпизоде, назначить расу владельца, но это будет немного странный и неудобный путь решения. Проще использовать передачу параметра в библиотечный Эпизод, как показано в приведенном ниже примере:

```
<cue name="mylibrarycue" library="1">
   <action>
      ...
      <create_ship ...="" race="{param@targetrace}" ...="" >
         ...
      </create_ship>
      ...
   </action>
</cue>
...
<cue name="myfirstcue">
   ...
   <cues>
      <cue ref="mylibrarycue">
         <params>
            <param name="targetrace" value="{value@myfirstcue.myrace}"/>
         </params>
      </cue>
   </cues>
</cue>
...
<cue name="mysecondcue">
   <condition>
      ...
   </condition>
   <action ref="mylibrarycue">
      <params>
         <param name="targetrace" value="{lookup.race@boron}"/>
      </params>
   </action>
</cue>
```

Обратите внимание, как в данном примере мы в одном случае используем библиотечный Эпизод целиком, в качестве вложенного Эпизода, и в другом случае мы используем только секцию Действий данного библиотечного Эпизода.

Поскольку параметр не конкретизирован, то в обоих случаях, при вызове библиотечного Эпизода будет использован параметр `"targetrace"`. Если вам требуется большая конкретизация того, в каких случаях какой параметр должен быть использован при вызове библиотечного Эпизода, в зависимости от того, как был вызван такой Эпизод, вам надо конкретизировать используемый параметр , к примеру, использовать `"{param.cue@targetrace}"` или `"{param.action@targetrace}"`, в этом случае значение переменной параметра будет передано только при вызове Эпизода в указанной секции, в нашем случае - Эпизода целиком, или, только секции Действий. Вы можете использовать столько параметров, сколько Вам нужно, и можете передавать через них как конкретные значения, к примеру `"1"` или `"m5"`, так и переменные. Вы можете описывать библиотечные

Эпизоды как на верхнем уровне иерархии вложенности в файле директора, так и внутри других Эпизодов, и тогда они будут доступны только Эпизодам лежащим на более низких уровнях иерархии вложенности в данный Эпизод. Для Эпизодов же верхнего уровня вложенности, библиотечные Эпизоды, описанные на этом же верхнем уровни, так же доступны. И самое, наконец, интересное, Эпизод не обязательно метить как библиотечный, для того чтобы использовать его в качестве библиотечного. Вы можете использовать любой Эпизод в качестве библиотечного, главное, чтобы он был доступен там, откуда его вызвали, согласно видимости по уровням иерархии вложенности.