

Запуск системы на Windows

Создание структуры директорий:

1. Создаем папку `empty_project` и копируем в нее файлы `start.exe` и все `dll`-файлы из дистрибутива.
2. Создаем папку `images` – каталог хранения рендеров, его мы укажем в конфигурационном файле в качестве `ImagesPath`. Необходимо поместить туда файл `logo.png`
3. Создаем папку `configs` – каталог для хранения конфигурационных файлов (кроме основного конфигурационного файла, который должен быть расположен в корневом каталоге)
4. Создаем папку `fonts` – каталог для хранения используемых шрифтов

Создание файлов проекта:

`config.xml`

Создаем `config.xml` в папке `empty_project`. `Config.xml` – файл, который платформа загружает по умолчанию. Если указать в качестве параметра запуска другой файл, то вместо `config.xml` будет загружен указанный. В тэге `System` определяем настройки программы. В тэге `VM` определяем виртуальную машину с её настройками. В тэге `Font` определяем используемый шрифт:

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<root>
  <System
    Language="ru"
    ImagesPath="images/"
    ConfigsPath="configs/"
    Log="true"/>
  <VM
    ID="main"
    Visual="panels.xml"
    StartupLayout="viewports.xml"
    StartupLogic="start.lua" >
  </VM>
  <Font ID="Default" FileName="fonts/LiberationSerif-Regular.ttf" />
</root>
```

`viewports.xml`

Создаем `viewports.xml` в папке `empty_project/configs/`, указанный в атрибуте `StartupLayout` файла `config.xml`. Это файл с настройками окон программы, в нем указывается разрешение окон программы (`MonitorWidth` и `MonitorHeight`) и монитор, на котором они будут отображены

(тэг ViewPort, атрибут Monitor). Система может оперировать несколькими мониторами. Определим одно окно программы на первом мониторе, разрешение – 800×480

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<root MonitorWidth="800" MonitorHeight="480">
  <Viewport Monitor="0,0" ID="main_viewport" />
</root>
```

panels.xml

Создаем panels.xml, указанный в атрибуте Visual файла config.xml. В файле panels.xml указываются элементы, которые будут использоваться в процессе работы программы. В тэге Text указываем файл с текстовыми ресурсами. В тэге Library указываем файл с шаблонами элементов. В тэге GUI создаем тег Panel с набором элементов на панели. Для каждого элемента необходимо определить тег Entry и задать ID элемента иTypeID из файла templates.xml.

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<root>
  <Text FileName="text.xml"/>
  <Library FileName="templates.xml"/>
  <GUI>
    <Panel ID="main" VPWidth="800" VPHeight="480">
      <Comment>Empty project main panel</Comment>
      <Entry ID="bg" TypeID="bg">
        <Comment>Panel background</Comment>
      </Entry>
      <Entry ID="btExit" TypeID="button" Top="300" Left="300"/>
      <Entry ID="Logo" TypeID="logo" Top="60" Left="60"/>
    </Panel>
  </GUI>
</root>
```

templates.xml

Создаем templates.xml. В этом файле определим шаблоны элементов, которые можно использовать для графических элементов. Каждый элемент может содержать различные типы подэлементов. В этой статье используются только типы Rectangle, TextBox и Image. Опишем в templates.xml все элементы, использованные в panels.xml. Rectangle – отрисовывается прямоугольник с заданными параметрами. TextBox – выводится текст. Image – выводится рендер из указанного файла.

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<root>
  <Library>
    <Element ID="bg" Width="800" Height="480">
      <Comment>Background</Comment>
      <Rectangle ID="bg" Rect="50,50,700,380" Color="white">
```

```

BGCOLOR="black" Thickness="1" CornerRadius="10"/>
    <TextBox ID="text" Rect="full" TextTID="txtEmptyProject"
Font="Default" Size="60" Align="Center" VAlign="Center"/>
</Element>
<Element ID="button" Width="200" Height="80">
    <Rectangle ID="bg" Rect="full" Color="white" BGCOLOR="black"
Thickness="2" CornerRadius="5"/>
    <TextBox ID="text" Rect="full" TextTID="txtExit" Font="Default"
Size="30" Align="Center" VAlign="Center"/>
</Element>
<Element ID="logo" Width="369" Height="595" Scale="0.25">
    <Image Rect="full" Source="logo.png"/>
</Element>
</Library>
</root>

```

text.xml

Создаем text.xml, указанный в качестве текстового ресурса в panels.xml. Платформа позволяет легко создавать мультиязычный интерфейс, для этого необходимо для каждого текста определить перевод в текстовых ресурсах.

```

<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<root>
    <Text ID="txtEmptyProject">
        <Entry Lang="ru" Value="Пустой проект" />
        <Entry Lang="en" Value="Empty project" />
    </Text>
    <Text ID="txtExit">
        <Entry Lang="ru" Value="Выход" />
        <Entry Lang="en" Value="Exit" />
    </Text>
</root>

```

start.lua

Создаем start.lua, указанный в тэге StartupLogic файла config.xml. После того как все графические элементы определены, необходимо прописать логику их использования. Это делается в файле, указанном в StartupLogic. Чтобы описанная нами панель main появилась в окне с ID="main_viewport", необходимо указать это в файле start.lua, следующим образом:

```
Panel('main_viewport', 'main')
```

На панели определена кнопка btExit, чтобы обработать событие нажатия на эту кнопку, необходимо прописать это в файле start.lua:

```

function H_Exit(_ev)
    sys.Exit()

```

```
end  
AddEventHandler('main.btExit.Mouse.LeftPress', 'H_Exit')
```

Полный текст файла start.lua:

```
Panel('main_viewport', 'main')  
  
function H_Exit(_ev)  
    sys.Exit()  
end  
AddEventHandler('main.btExit.Mouse.LeftPress', 'H_Exit')
```

Запуск

[Запускаем](#) start.exe. Таким образом мы описали одно окно с разрешением 800x480, расположенное на первом мониторе. В этом окне отрисовано три элемента – бэкграунд с текстом, рендер с логотипом и кнопка выхода:



[empty_project.zip](#)

From:

<https://wiki.lambda-mu.com/> - **Lambda-Mu Wiki**

Permanent link:

<https://wiki.lambda-mu.com/lm2/ce/winstart?rev=1583987159>

Last update: **2020/11/30 14:12**

