Рыбинский государственный авиационный технический университет им. П.А.Соловьева

Факультет радиоэлектроники и информатики

Кафедра математического и программного обеспечения электронных вычислительных средств

Лабораторная работа

по дисциплине

Введение в программную инженерию

Студент группы ИПБ-16 Рахимов К.С.

Руководитель Воробьев К.А.

Рыбинск 2017г.

Оглавление

[1. Архитектура приложения 2](#_Toc483857473)

[2. JSON 3](#_Toc483857474)

[3. HTML5 + CSS3 4](#_Toc483857475)

[4. JavaScript 8](#_Toc483857476)

# Архитектура приложения

Kamych Editor/

└──KED/

├── CSS/

└── main.css

│

├── img/

│

├── js/

└── jscolor/

└── jscolor.js

└── jquery-1.11.2.js

└── main.js

│

├── index.html

│

├── KED.exe

│

├── package.json

Данная программа собрана и разработана на среде “Node-webkit” (позволяющая создавать кроссплатформенные приложения с интерфейсом), то есть простыми словами “Node-webkit” делает из сайта десктопное приложение, чтобы построить такое десктопное приложение потребуется:

1. JSON
2. HTML5 + CSS3
3. JavaScript

На этих языках/формальных языках построена данная программа, ниже будут описаны интеграции всех этих языков с другом-другом.

# JSON

Каждый манифест (package.json) приложения обязан предоставить следующие поля:

* **name** (string) — Уникальное имя приложения, предоставленное в виде name-of-application. Может включать в себя следующие символы: .\_-.
* **main** (string) — Страница, которая будет открыта при запуске приложения.

Далее будут описаны только те функции, которые использованы в данной программе:

“name”: “KED”,

“main”: “index.html”,

“Window”: {

“width”: 800,

“height”: 600,

“icon”: “img/icon.ico”

}

* window – главное окно, в данном поле содержатся подполя
  + width/height (int) – начальная ширина/высота окна
  + icon (string) – путь до иконки (логотипа) приложения

# HTML5 + CSS3

Выше мы только создали десктопное окно, но чтобы был какой либо контент, в проекте должен быть шаблон из HTML и стилей CSS.

В данном приложение используется HyperText Markup Language пятой версии (index.html) и Cascading Style Sheets третей версии (main.css)

Основное из файла ”index.html”:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<link rel="stylesheet" href="/css/main.css">

<script src="/js/jscolor/jscolor.js"></script>

</head>

<body>

<!-- First Container -->

<div class="f\_block">

//content

</div>

<!-- Second container -->

<div class="s\_block" id="s\_block">

<div class="s\_block\_right">

<div class="block"></div>

<textarea style="color: black; width: 300px; height: 105px; resize: none; overflow: hidden;" id="code-result" readonly></textarea>

</div>

<div class="s\_block\_left">

<button class="btn" id="increase-width">↑</button>

<button class="btn" id="reduse-width">↓</button>

<input type="number" id="step-block-width" class="input" min="0" max="380" value="1"></input>

<button class="btn" id="increase-height">↑</button>

<button class="btn" id="reduse-height">↓</button>

<input type="number" id="step-block-height" min="0" max="300" class="input" value="1"></input>

<button class="btn" id="increase-radius">↑</button>

<button class="btn" id="reduse-radius">↓</button>

<input type="number" id="step-block-br" min="0" max="60" class="input" value="1"></input>

<input type="text" value="000000" class="input\_bg jscolor" id="border-color">

<input type="text" value="FFFFFF" class="input\_bg jscolor" id="bg-color">

<!-- Third container -->

<div class="t\_block">

//content

</div>

</body>

</html>

<script src="/js/jquery-1.11.2.js"></script>

<script src="/js/main.js"></script>

Основное из файла ”main.css”:

body{

margin: 0;

padding: 0;

overflow-x: hidden;

}

.f\_block{

height: 100vh;

width: 100%;

background: black;

}

.s\_block\_right{

width: 50%;

height: 100vh;

float: left;

display: inline-block;

}

.block{

margin: 0 auto;

width: 200px;

height: 100px;

background: #FFFFFF;

border: 5px solid #000000;

border-radius: 0px;

}

.s\_block\_left{

width: 50%;

height: 100vh;

background: black;

float: right;

display: inline-block;

}

.btn{

font: 0.5em Tahoma;

background: black;

color: white;

width: 50px;

height: 20px;

outline: none;

resize: none;

border: 1px solid white;

margin-left: 10px;

}

.btn:hover{

cursor: pointer;

}

.input{

width: 100px;

background: black;

resize: none;

outline: none;

color: white;

border-top: none;

border-left: none;

border-right: none;

border-bottom: 1px solid white;

}

.input\_bg{

resize: none;

outline: none;

color: white;

border-top: none;

border-left: none;

border-right: none;

border-bottom: 1px solid white;

width: 32%;

}

.t\_block{

height: 20vh;

}

# JavaScript

Так как идея данного приложения это реализация обработки CSS блока в реальном времени, то просто шаблоном из HTML и СSS не обойтись, для этого нужен функционал. JavaScript – в помощь!

Весь функционал находится в файле “main.js”.

Для того чтобы иметь связь с JavaScript’ом и HTML5, нужно прописать переменные в JavaScript'е и закэшировать кнопку или форму, которые находятся в HTML5 (<button>, <Input>).

1. block = $('.block') – блок над котором будут проведены действия,
2. increaseRad = $('#increase-radius') – кнопка увеличения радиуса,
3. reduseRad = $('#reduse-radius') – кнопка уменьшения радиуса,
4. maxRadius = 60 – константа , максимальный радиус блока,
5. minRadius = 0 - константа, минимальный радиус блока,
6. increaseWid = $('#increase-width') - кнопка увеличения ширины,
7. reduseWid = $('#reduse-width') – кнопка уменьшения ширины,
8. maxWid = 380 - константа , максимальная ширина блока,
9. minWid = 10 - константа , минимальная ширина блока,
10. increaseHeig = $('#increase-height') – кнопка увеличения ширины,
11. reduseHeigh = $('#reduse-height') – кнопка уменьшения ширины,
12. maxHeig = 300 - константа , максимальная высота блока,
13. minHeig = 10 - константа , минимальная высота блока,
14. bgColorInput = $('#bg-color') – форма для изменения фонового-цвета блока,
15. brColorInput = $('#border-color') – форма для изменения цвета обводки блока,
16. codeResultArea = $('#code-result') – поле для хранения CSS3 кода.

* increaseWid.on('click', function(){…}) - Обработчик события, который прослушивает событие ‘click’ по кнопке увеличения ширины.
* reduseWid.on('click', function(){…}) - Обработчик события, который прослушивает событие ‘click’ по кнопке уменьшения ширины.
* increaseHeig.on('click', function(){…}) - Обработчик события, который прослушивает событие ‘click’ по кнопке увеличения высоты.
* reduseHeigh.on('click', function(){…}) - Обработчик события, который прослушивает событие ‘click’ по кнопке уменьшения высоты.
* increaseRad.on('click', function(){…}) - Обработчик события, который прослушивает событие ‘click’ по кнопке увеличения радиуса обводки.
* reduseRad.on('click', function(){…}) – Обработчик события, который прослушивает событие ‘click’ по кнопке уменьшения радиуса обводки.
* bgColorInput.on('change', function(){…}) - Обработчик события, который прослушивает событие ‘click’ по изменению фонового цвета.
* brColorInput.on('change', function(){…}) - Обработчик события, который прослушивает событие ‘click’ по изменению фонового цвета обводки.
* var updateResult = function(){…}) – функция которая вычисляет текущее значение блока, она должна быть размещена в каждой функции обработчика в виде:
  + updateResult();