

ВИСОКА ШКОЛА ЕЛЕКТРОТЕХНИКЕ И РАЧУНАРСТВА  
СТРУКОВНИХ СТУДИЈА

**Палијан Иван**

**Имплементација генератора кода за веб апликациони  
сервер у програмском језику *Java***

**- завршни рад -**



Београд, септембар 2017.

Кандидат: **Палијан Иван**

Број индекса: **ЕЛИТЕ-11/14**

Студијски програм: **Електроника и телекомуникације**

Тема: **Имплементација генератора кода за веб апликациони сервер у програмском језику *Java***

Основни задаци:

1. Опис елемената **Java-е** и **SPRING-а** коришћених за реализацију рада.
2. Имплементација генератора кода за веб апликациони сервер.
3. Опис корисничког интерфејса са примером у **SPRING** технологији.

Ментор:

Београд, септембар 2017.

---

др. Перица Штрбац, проф. ВИШЕР



## РЕЗИМЕ:

У овом раду је представљена *ILib* библиотека као и пратећи демо пројекат из кога корисници могу тачно да виде примену ове библиотеке као и како могу сами да се прикључе развоју исте или како је могу применити у својим пројектима. Постоји демо страница *index* где је приказано како се користи библиотека. *Device* страница садржи код за контролу уређаја.

Библиотека као и пробни пример реализовани су у програмском језику *Java* док је *embedded* део направљен у *C*-у.

## ABSTRACT:

This work presents ILib library and accompanying demo project from which users can gain insight into workings of this library, how to use it and how to contribute its development. There is accompanying demo page index that is used as manual for user to see how to initialize library. Device page is used for communication and control of microcontroller.

Library and demo example are realized in Java while embedded parts are written in C.

## САДРЖАЈ:

1.	Увод	1
2.	Кориштене технологије	2
2.1.	<i>Java</i>	2
2.1.1.	Увод	2
2.1.2.	Писање и извршавање Јава кода	2
2.1.3.	Уништавање објеката	2
2.2.	Spring Framework	3
2.2.1.	Увод	3
2.3.	Програмски језик “С”	3
2.3.1.	Увод	3
2.3.2.	Настанак	3
2.4.	Развојно окружење	4
2.5.	Кориштене Spring а anotације	4
2.6.	<b>File</b> структура	5
3.	Код класе ILib_p	7
4.	Код класе ILib_CSSStyle	9
5.	Код странице index	15
6.	<i>Embedded</i> део	18
7.	ЗАКЉУЧАК	22
8.	ЛИТЕРАТУРА	23

## 1. УВОД

После кориштења великог броја алата аутор је увидео недостатке истих и одлучио је да направи библиотеку која ће омогућити програмерима који пишу у програмском језику *Java* да могу да креирају апликације које имају веб интерфејс и могу да приступају физичким компонентама система без учења додатних језика са *Java POJO (Plain Old Java Object)* објектима, што за корисника значи да може да почне да пројектује програм у јави и кад се појави потреба за веб интерфејсом врло лако може да дода исти.

Уз ову библиотеку иде и демо програм који показује примену ове библиотеке у *embedded* систему где *Raspberry Pi 2 model B* подиже веб сервер, *PI4J* библиотека читава температуру процесора и преко *UART*-а комуницира са *NUCLEO-F767ZI* развојном плочом.

## 2. КОРИШТЕНЕ ТЕХНОЛОГИЈЕ

У изради овог пројекта корштено је више технологија и језика. Примарне технологије су наведене у следећим поглављима.

### 2.1. JAVA

#### 2.1.1. Увод

Јава (енгл. *Java*, изговор: јава, \*џава) је објектно-оријентисани програмски језик, који је развила компанија Sun Microsystems почетком 1990-их година. Многи концепти Јаве су засновани на језику Оберон, Никлауса Вирта, творца Паскала, Модуле и других језика, и Ханспетера Месенбека. Избацили су концепт модула и увели пакете какве данас знамо, који се ослањају на фајл систем и увели формално концепт класа из објектно-оријентисане парадигме. Осим тога, језик има синтаксу сличну језицима С и С++, али је много строжи при превођењу, дизајниран тако да буде независан од платформе, и са поједностављеним управљањем меморијом. Претпоставља се да је ово урађено због популарности језика С, али и због једноставности неких структура. Прва верзија је званично објављена 1995. године. [1]

#### 2.1.2. Писање и извршавање Јава кода

Јава је објектно-оријентисани програмски језик те као такав поштује правило да се једна класа налази у једном фајлу (осим унутрашњих класа). Изворни код се чува у фајловима са наставком .java. Програми написани у програмском језику Јава се не компајлирају у машински код, већ се превode у бајт-код, и тако преведени фајлови имају наставак .class. Да би се извршио програм написан у Јави, неопходно је имати Јава Виртуелну Машину, на којој се интерпретира бајт-код. Управо се кориштењем Јава Виртуелне Машине постиже независност од платформе, тако да се исти бајт-код може једнако извршавати на сваком оперативном систему на коме је инсталирана Јава Виртуелна Машина.[1]

#### 2.1.3. Уништавање објеката

Уклањање непотребних објеката из меморије обавља garbage collector. Овај процес ради независно од покренутих програма и самостално одлучује које објекте ће уклонити из меморије. Осим уклањања сувишних објеката он врши дефрагментацију меморије. Његовим кориштењем се програмер ослобађа посла који се односи на ослобађање меморије кроз програмски код. На овај начин се не мора водити рачуна о деструкторима већ се њихов посао изводи аутоматски. Garbage collector је могуће покренути и ручно. [1]

## 2.2. SPRING FRAMEWORK

### 2.2.1. Увод

Spring Framework је апликациони framework и container са инверзијом контроле за Java платформу. Функције овог framework-а може користити било која Java апликација али постоје екстензије за прављење веб апликација које се заснивају на Java EE (Enterprise Edition) платформи. Иако овај framework не захтева специфичан програмски модел постао је веома популаран као алтернатива Enterprise JavaBeans (EJB) моделу. Spring Framework је open source. [2]

## 2.3. ПРОГРАМСКИ ЈЕЗИК “C”

### 2.3.1. Увод

Програмски језик C (чита се: "це" - C је латинично слово) је процедурални програмски језик, настао 1972. године. Аутор језика је Денис Ричи, а настао је у истраживачком центру Bell Laboratories у Њу Џерзију за потребе оперативног система UNIX. [3]

### 2.3.2. Настанак

Прве верзије оперативног система UNIX су биле, због неопходне брзине извршавања, писане користећи асемблер, али се овај пројекат брзо приближавао свом крају јер је програмски код асемблера изразито велик и тежак за одржавање. Архитектама UNIX-а је био потребан програмски језик који би задржао ефикасност асемблера, али истовремено пружио и комфор већ постојећих виших програмских језика, попут Паскала, Кобола и др. Идеја је била начинити програмски језик који би имао све ово, али чија архитектура и логика не би много одударала од асемблерске. Због тога C-ове контролне структуре наликују на асемблерске, а постоје и изрази који директно позивају одговарајуће асемблерске инструкције. Тако, израз `x++` директно позива асемблерску инструкцију `INC`. Језик је био потпун успех, и користио се за писање свих будућих верзија UNIX-а. У градњи пројекта је учествовао као главни пројектант Денис Ричи, који је уосталом био и један од водећих пројектаната за оперативни систем UNIX. Програмски језик C је језик опште намене. Иако је развијан као „језик за системско програмирање“, подједнако добро се користи за писање значајних програма у различитим областима. C је такође веома утицао на многе друге популарне програме, посебно на C++, који је оригинално развијан као надградња C-а. [3]



## 2.4. РАЗВОЈНО ОКРУЖЕЊЕ

Језици и развојна окружења коришћени за овај пројекат су следећи:

- Java 1.8
- C
- Maven 4.0.0
- Spring 2.0.0.M3
- MongoDB driver 4.1.3
- pi4j 1.1
- JetBrains IntelliJ IDEA 2017.2.4
- KEIL

## 2.5. КОРИШТЕНЕ SPRING АНОТАЦИЈЕ

### `@SpringBootApplication`

Ова анотација се користи као замена за следећу групацију:

`@Configuration` Индикује да класа садржи једну или више `@bean` метода.

`@EnableAutoConfiguration` Омогућава ауто конфигурацију.

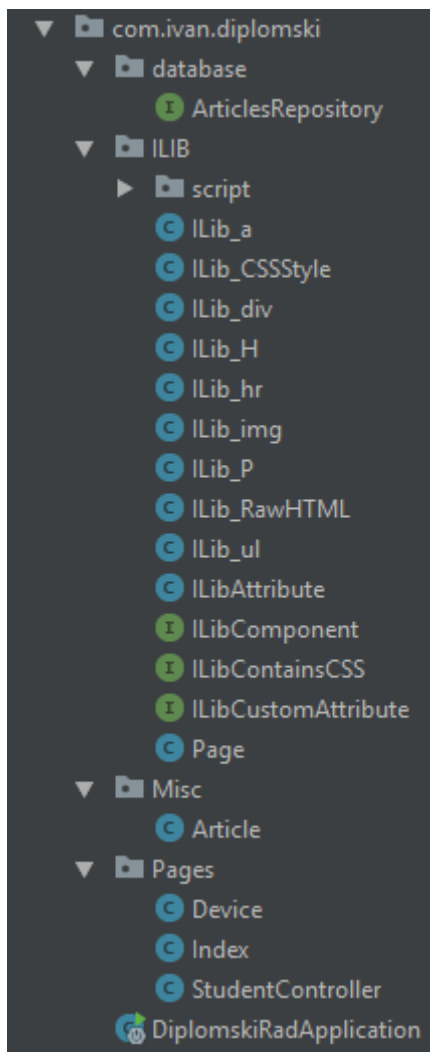
`@ComponentScan` Ова анотација говори spring-у и којим пакетима се налазе класе са spring анотацијам које треба да буду контролисане од стране spring-a

Због честог коришћења ових анотација оне се замењују једном.

`@ServletComponentScan` Омогућава скенирање у потрази за Servlet компонентама.

`@RequestMapping(value = "/index", method = RequestMethod.GET)` Омогућава мапирање веб адресе на функцију као и метод упита на који ће функција бити повезана.

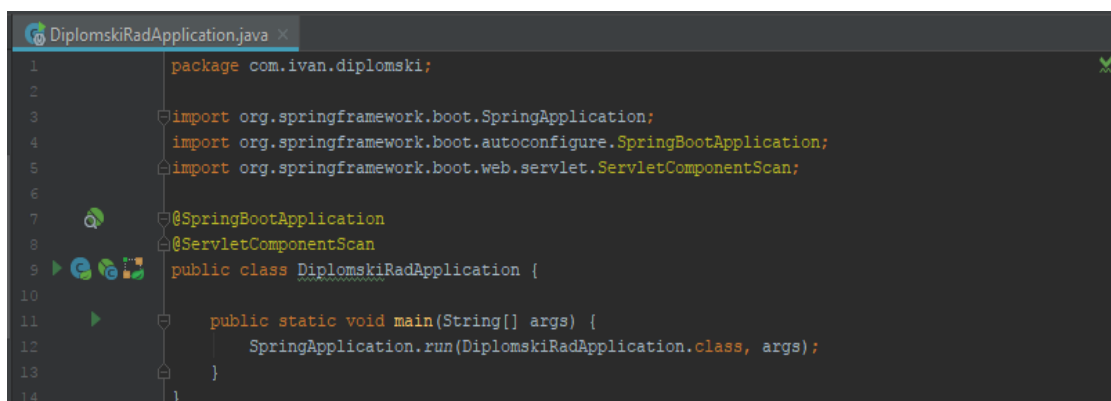
## 2.6. FILE СТРУКТУРА



Слика 1. – *File* структура

На слици 1 је приказана структура изворног кода демонстрационе апликације са увезеним кодом за генератор веб елемената у даљем тексту *ILib*.

У *ILib* пакету се налазе класе које могу да се инстанцирају и представљају истоимене *HTML* елементе у формату *ILib\_XXX* где *XXX* представља име *HTML tag*.



```
1 package com.ivan.diplomski;
2
3 import org.springframework.boot.SpringApplication;
4 import org.springframework.boot.autoconfigure.SpringBootApplication;
5 import org.springframework.boot.web.servlet.ServletComponentScan;
6
7 @SpringBootApplication
8 @ServletComponentScan
9 public class DiplomskiRadApplication {
10
11     public static void main(String[] args) {
12         SpringApplication.run(DiplomskiRadApplication.class, args);
13     }
14 }
```

Слика 2. – Класа *DiplomskiRadApplication*

На слици 2 је приказана класа *DiplomskiRadApplication* то је класа која садржи улазну функцију *Main* и покреће *Spring* сервер.

Све класе које представљају тагове имплементирају од један до три од следећих интерфејса:

-*ILibComponent* -> Овај интерфејс означава да класа која га имплементира може да се третира као компонента.

-*ILibContainsCSS* -> Садржи функције које се користе за додавање *CSS* стилова елементу као и функције потребне за позадинско генерисање кода потребног да омогући ову функционалност. Ова библиотека подржава стилове дефинисане *CSS*-ом.

-*ILibCustomAttribute* -> Овај интерфејс је додат у једном од новијих *build*-ова и омогућава *RAW* мод. Што значи да може да се дода било који атрибут елементу подржан од стране *HTML*-а.

### 3. КОД КЛАСЕ ILib\_P

Следећи код представља имплементацију *HTML* `<p>` тага.

Из овог кода видимо да класа *ILib\_P* имплементира сва три интерфејса. Промењива типа *string* са називом *text* означава садржај овог параграфа. *ArrayList*-а *CSSElements* садржи све појединачне *CSS* елементе који делују на овај параграф. Листа под називом *RawAttributes* садржи атрибуте везане за *HTML* елемент. Интерфејс *ILibCustomAttribute* захтева имплементацију следећих метода *AddCustomAttribute* и *PrintCustomAttribute*. Прва метода служи да корисник дода атрибут елементу док друга служи за генерисање кода у позадини. . Интерфејс *ILibContainsCSS* захтева имплементацију следећих метода *AddCSSStyle* и *ReadCSSStyles*. Прва метода служи да корисник дода атрибут елементу док друга служи за генерисање кода у позадини. Метода *toString* извршава коначно паковање кода и исписивање.

```
-----  
package com.ivan.diplomski.ILIB;  
  
import java.util.ArrayList;  
  
/**  
 * Created by Ivan Palijan on 7/16/2017.  
 */  
public class ILib_P implements ILibComponent, ILibCustomAttribute, ILibContainsCSS  
{  
    String text;  
  
    ArrayList<ILib_CSSStyle> CSSElements = new ArrayList<>();  
    ArrayList<ILibAttribute> RawAttributes = new ArrayList<>();  
  
    @Override  
    public void AddCustomAttribute(String attrib, String value) {  
        RawAttributes.add(new ILibAttribute(attrib, value));  
    }  
  
    @Override  
    public String PrintCustomAttribute() {  
        StringBuilder sb = new StringBuilder();  
  
        for (ILibAttribute item : RawAttributes)  
        {  
            sb.append(" ").append(item.AttributeName).append(" =  
").append "\""").append(item.AttributeValue).append("\" " ");  
        }  
        return sb.toString();  
    }  
  
    @Override  
    public void AddCSSStyle(ILib_CSSStyle style) {  
        CSSElements.add(style);  
    }  
}
```

```
@Override
public String ReadCSSStyles() {
    StringBuilder sb = new StringBuilder();
    sb.append("style=\"");
    for (ILib_CSSStyle item : CSSElements)
    {
        sb.append(item.toString());
    }
    sb.append("\");
    return sb.toString();
}

public ILib_P(String text) {
    this.text = text;
}

public String getText() {
    return text;
}

public void setText(String text) {
    this.text = text;
}

@Override
public String toString() {
    return "<p " + ReadCSSStyles() + PrintCustomAttribute() + ">" + text +
"</p>";
}
}
```

## 4. КОД КЛАСЕ ILIB\_CSSSTYLE

Класа *ILIB\_CSSSTYLE* се користи за додавање стилова елементима. Поред конструктора и *override*-ованог *toString* у класи се налази и један *enum* са називом *CSSStyles* у њему се налазе сви *properties* дефинисани CSS-ом.

Пример коришћења објекта класе *ILIB\_CSSSTYLE* на слици 3:

```
test.AddCSSStyle(new ILib_CSSStyle(ILib_CSSStyle.CSSStyles.color.toString(),
„red"));
test.AddCSSStyle(new ILib_CSSStyle(ILib_CSSStyle.CSSStyles.font_size.toString(),
„50px"));
test.AddCSSStyle(new
ILib_CSSStyle(ILib_CSSStyle.CSSStyles.background_color.toString(), „blue"));
```

Слика 3. – Пример коришћења *ILIB\_CSSSTYLE* класе

---

```
package com.ivan.diplomski.ILIB;

public class ILib_CSSStyle
{
    String Style;
    String Value;

    public ILib_CSSStyle(String style, String value) {
        Style = style;
        Value = value;
    }

    @Override
    public String toString() {
        return this.Style + " : " + this.Value + "; ";
    }

    //style="color:blue; background-color: powderblue;"
    public enum CSSStyles
    {
        //https://www.w3schools.com/cssref/default.asp

        /**
         * Color Properties
         */
        color("color"),
        opacity("opacity"),

        /**
         * Background and Border Properties
         */
        background("background"),
        background_attachment("background-attachment"),
        background_blend_mode("background-blend-mode"),
        background_color("background-color"),
        background_image("background-image"),
        background_position("background-position"),
        background_repeat("background-repeat"),
        background_clip("background-clip"),
        background_origin("background-origin"),
        background_size("background-size"),
        border("border"),
        border_radius("border-radius"),
        border_bottom("border-bottom"),
        border_bottom_color("border-bottom-color"),
        border_bottom_left_radius("border-bottom-left-radius"),
```

```
border_top_right_radius("border-top-right-radius"),
border_top_style("border-top-style"),
border_top_width("border-top-width"),
border_width("border-width"),
box_decoration_break("box-decoration-break"),
box_shadow("box-shadow"),
```

```
/**
 * Basic Box Properties
 */
bottom("bottom"),
clear("clear"),
clip("clip"),
display("display"),
float_("float"),
height("height"),
left("left"),
margin("margin"),
margin_bottom("margin-bottom"),
margin_left("margin-left"),
margin_right("margin-right"),
margin_top("margin-top"),
max_height("max-height"),
max_width("max-width"),
min_height("min-height"),
min_width("min-width"),
overflow("overflow"),
overflow_x("overflow-x"),
overflow_y("overflow-y"),
padding("padding"),
padding_bottom("padding-bottom"),
padding_left("padding-left"),
padding_right("padding-right"),
padding_top("padding-top"),
position("position"),
right("right"),
top("top"),
visibility("visibility"),
width("width"),
vertical_align("vertical-align"),
z_index("z-index"),
```

```
/**
 * Flexible Box Layout
 */
align_content("align-content"),
align_items("align-items"),
align_self("align-self"),
flex("flex"),
flex_basis("flex-basis"),
flex_direction("flex-direction"),
flex_flow("flex-flow"),
flex_grow("flex-grow"),
flex_shrink("flex-shrink"),
flex_wrap("flex-wrap"),
justify_content("justify-content"),
order("order"),
```

```
/**
 * Text Properties
 */
hanging_punctuation("hanging-punctuation"),
hyphens("hyphens"),
letter_spacing("letter-spacing"),
line_break("line-break"),
line_height("line-height"),
```

```
overflow_wrap("overflow-wrap"),
tab_size("tab-size"),
text_align("text-align"),
text_align_last("text-align-last"),
text_combine_upright("text-combine-upright"),
text_indent("text-indent"),
text_justify("text-justify"),
text_transform("text-transform"),
white_space("white-space"),
word_break("word-break"),
word_spacing("word-spacing"),
word_wrap("word-wrap"),

/**
 * Text Decoration Properties
 */
text_decoration("text-decoration"),
text_decoration_color("text-decoration-color"),
text_decoration_line("text-decoration-line"),
text_decoration_style("text-decoration-style"),
text_shadow("text-shadow"),
text_underline_position("text-underline-position"),

/**
 * Font Properties
 */
font_face("@font-face"),
font_feature_values("@font-feature-values"),
font("font"),
font_family("font-family"),
font_feature_settings("font-feature-settings"),
font_kerning("font-kerning"),
font_language_override("font-language-override"),
font_size("font-size"),
font_size_adjust("font-size-adjust"),
font_stretch("font-stretch"),
font_style("font-style"),
font_synthesis("font-synthesis"),
font_variant("font-variant"),
font_variant_alternates("font-variant-alternates"),
font_variant_caps("font-variant-caps"),
font_variant_east_asian("font-variant-east-asian"),
font_variant_ligatures("font-variant-ligatures"),
font_variant_numeric("font-variant-numeric"),
font_variant_position("font-variant-position"),
font_weight("font-weight"),

/**
 * Writing Modes Properties
 */
direction("direction"),
text_orientation("text-orientation"),
unicode_bidi("unicode-bidi"),
user_select("user-select"),
writing_mode("writing-mode"),

/**
 * Table Properties
 */
border_collapse("border-collapse"),
border_spacing("border-spacing"),
caption_side("caption-side"),
empty_cells("empty-cells"),
table_layout("table-layout"),

/**
```



```
* Lists and Counters Properties
*/
counter_increment("counter-increment"),
counter_reset("counter-reset"),
list_style("list-style"),
list_style_image("list-style-image"),
list_style_position("list-style-position"),
list_style_type("list-style-type"),

/**
 * Animation Properties
 */
keyframes("@keyframes"),
animation("animation"),
animation_delay("animation-delay"),
animation_direction("animation-direction"),
animation_duration("animation-duration"),
animation_fill_mode("animation-fill-mode"),
animation_iteration_count("animation-iteration-count"),
animation_name("animation-name"),
animation_play_state("animation-play-state"),
animation_timing_function("animation-timing-function"),

/**
 * Transform Properties
 */
backface_visibility("backface-visibility"),
perspective("perspective"),
perspective_origin("perspective-origin"),
transform("transform"),
transform_origin("transform-origin"),
transform_style("transform-style"),

/**
 * Transitions Properties
 */
transition("transition"),
transition_property("transition-property"),
transition_duration("transition-duration"),
transition_timing_function("transition-timing-function"),
transition_delay("transition-delay"),

/**
 * Basic User Interface Properties
 */
box_sizing("box-sizing"),
content("content"),
cursor("cursor"),
ime_mode("ime-mode"),
nav_down("nav-down"),
nav_index("nav-index"),
nav_left("nav-left"),
nav_right("nav-right"),
nav_up("nav-up"),
outline("outline"),
outline_color("outline-color"),
outline_offset("outline-offset"),
outline_style("outline-style"),
outline_width("outline-width"),
resize("resize"),
text_overflow("text-overflow"),

/**
 * Multi-column Layout Properties
 */
break_after("break-after"),
```

```
break_before("break-before"),
break_inside("break-inside"),
column_count("column-count"),
column_fill("column-fill"),
column_gap("column-gap"),
column_rule("column-rule"),
column_rule_color("column-rule-color"),
column_rule_style("column-rule-style"),
column_rule_width("column-rule-width"),
column_span("column-span"),
column_width("column-width"),
columns("columns"),
widows("widows"),

/**
 * Paged Media
 */
orphans("orphans"),
page_break_after("page-break-after"),
page_break_before("page-break-before"),
page_break_inside("page-break-inside"),

/**
 * Generated Content for Paged Media
 */
marks("marks"),
quotes("quotes"),

/**
 * Filter Effects Properties
 */
filter("filter"),

/**
 * Image Values and Replaced Content
 */
image_orientation("image-orientation"),
image_rendering("image-rendering"),
image_resolution("image-resolution"),
object_fit("object-fit"),
object_position("object-position"),

/**
 * Masking Properties
 */
mask("mask"),
mask_type("mask-type"),

/**
 * Speech Properties
 */
mark("mark"),
mark_after("mark-after"),
mark_before("mark-before"),
phonemes("phonemes"),
rest("rest"),
rest_after("rest-after"),
rest_before("rest-before"),
voice_balance("voice-balance"),
voice_duration("voice-duration"),
voice_pitch("voice-pitch"),
voice_pitch_range("voice-pitch-range"),
voice_rate("voice-rate"),
voice_stress("voice-stress"),
voice_volume("voice-volume"),
```

```
/**
 * Marquee Properties
 */
marquee_direction("marquee-direction"),
marquee_play_count("marquee-play-count"),
marquee_speed("marquee-speed"),
marquee_style("marquee-style");

private String attribute;

private CSSStyles(String attribute) {
    this.attribute = attribute;
}

@Override
public String toString() {
    return attribute;
}
}
```

## 5. КОД СТРАНИЦЕ INDEX

Страница *index* је демонстрациона страница *ILib* библиотеке која приказује основне могућности и представља подсетник при коришћењу библиотеке.

*Article* је класа која се користи у раду са *mongodb* базом класа садржи два поља за наслов и текст чланка и служи да покаже како се повезује библиотека са базом.

*ILib\_div* представља `<div>` таг и у библиотеци се често користи за складиштење података добијених преко AJAX-а.

*Page* је класа из које се генеришу објекти који представљају појединачне странице. Сваки објекат *Page* класе има методу *addComponent* која омогућава додавање објеката на страницу.

*Script AJAX* класа представља *AJAX* скрипту при позиву се прослеђују два аргумента први је *id* елемента у који ће се при повратку из методе уписати одговор сервера, а други аргумент представља адресу где се упућује захтев. Ппимер инстанцирања објекта класе *Script AJAX*

```
Script AJAX div11 = new Script AJAX("div1", "/device/OK");
```

Због велике количине различитих могућности ЈаваСЦРИПТА уведен је РАВВ резим где је омогућено директно писање ЈАВАСЦРИПТ кода.

```
stranica.StandardScripts.add(new scriptRaw("var myVar = setInterval(divTemp, 1000);));
```

Следећи код приказује коришћење `;a`: тага као и додавање позива функције *fundiv12.functionLink()* кад се притисне на овај елемент

```
ILib_a dugmeOFF = new ILib_a();  
dugmeOFF.hm.put(ILib_a.Attribute.ONCLICK, fundiv12.functionLink());  
dugmeOFF.setFieldText("OFF");  
stranica.addComponent(dugmeOFF);
```

Следећи код приказује рад са сликама:

```
ILib_img img = new ILib_img();  
img.AddCustomAttribute("src", "http://www.mybligr.com/wp-  
content/uploads/2017/03/cute-and-lovable-pictures-of-rabbits-2.jpg");  
img.AddCustomAttribute("alt", "rabbit");  
img.AddCustomAttribute("height", "300");  
img.AddCustomAttribute("width", "400");  
img.AddCSSStyle(new ILib_CSSStyle(ILib_CSSStyle.CSSStyles.border_radius.toString(),  
"50%"));  
stranica.addComponent(img);
```

---

```
package com.ivan.diplomski.Pages;

import com.ivan.diplomski.ILIB.*;
import com.ivan.diplomski.ILIB.script.Script AJAX;
import com.ivan.diplomski.ILIB.script.scriptFunction;
import com.ivan.diplomski.ILIB.script.scriptRaw;
import com.ivan.diplomski.Misc.Article;
import com.ivan.diplomski.database.ArticlesRepository;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RestController;

import java.util.List;

@RestController
@RequestMapping("/")
public class Index {

    @Autowired
    private ArticlesRepository hotelRepository;

    @RequestMapping(value = "/index", method = RequestMethod.GET)
    public String ika() {

        Article prvi = new Article("Naslov prvog clanka", "Ovaj clanak je povucen  
iz mongoDB baze");
        Article drugi = new Article("Naslov drugog clanka", "Ovaj clanak je kao i  
prethodni ucitan je iz mongoDB i sacinjen od naslova i teksta");

        this.hotelRepository.deleteAll();
        this.hotelRepository.save(prvi);
        this.hotelRepository.save(drugi);

        List<Article> povuceno = this.hotelRepository.findAll();

        Page stranica = new Page();

        ILib_div divTemp = new ILib_div("divTemp");
        divTemp.innerHTML = new ILib_H(1, "Core tmperature").toString();

        ILib_div div1 = new ILib_div("div1");
        div1.innerHTML = new ILib_H(1, "Molimo pritisnite dugme").toString();

        stranica.addComponent(divTemp);
        stranica.addComponent(div1);

        Script_AJAX div11 = new Script_AJAX("div1", "/device/OK");
        scriptFunction fundiv11 = new scriptFunction("ajax1", div11.toString());

        Script_AJAX div12 = new Script_AJAX("div1", "/device/NO");
        scriptFunction fundiv12 = new scriptFunction("ajax2", div12.toString());

        Script_AJAX divAjaxTemp = new Script_AJAX("divTemp", "/device/GET");
        scriptFunction fundivTemp = new scriptFunction("divTemp",
        divAjaxTemp.toString());

        stranica.StandardScripts.add(new scriptRaw("var myVar =  
setInterval(divTemp, 1000);"));
        stranica.scripts.add(fundiv11);
```

```
stranica.scripts.add(fundiv12);
stranica.scripts.add(fundivTemp);

ILib_a dugmeON = new ILib_a();
dugmeON.hm.put(ILib_a.Attribute.ONCLICK, fundiv11.functionLink());
dugmeON.setFieldText("ON");
stranica.addComponent(dugmeON);

ILib_a dugmeOFF = new ILib_a();
dugmeOFF.hm.put(ILib_a.Attribute.ONCLICK, fundiv12.functionLink());
dugmeOFF.setFieldText("OFF");
stranica.addComponent(dugmeOFF);

stranica.addComponent(new ILib_P("Dobro dosli na demonstraciju ILib
paketa"));

ILib_a dugme = new ILib_a();
dugme.hm.put(ILib_a.Attribute.ID, "ika");
dugme.hm.put(ILib_a.Attribute.NAME, "IME");
dugme.hm.put(ILib_a.Attribute.HREF, "/students/page2");
dugme.setFieldText("Page 2");

for (Article item : povuceno)
{
    stranica.addComponent(new ILib_H(1, item.Heading));
    stranica.addComponent(new ILib_P(item.text));
}

stranica.addComponent(new ILib_hr());

stranica.addComponent(new ILib_P("Sledeci element je tabela"));

ILib_ul temp = new ILib_ul();
for (Article item : povuceno)
{
    temp.AddNewElement(new ILib_H(1, item.Heading));
}
stranica.addComponent(temp);

ILib_P test = new ILib_P("Ovo je primer paragrafa koji je sacinjen od <p>
tagova i na njemu su primenjeni CSS stilovi");
test.AddCSSStyle(new
ILib_CSSStyle(ILib_CSSStyle.CSSStyles.color.toString(), "red"));
test.AddCSSStyle(new
ILib_CSSStyle(ILib_CSSStyle.CSSStyles.font_size.toString(), "50px"));
test.AddCSSStyle(new
ILib_CSSStyle(ILib_CSSStyle.CSSStyles.background_color.toString(), "blue"));
stranica.addComponent(test);
stranica.addComponent(dugme);
stranica.addComponent(new ILib_hr());

ILib_img img = new ILib_img();
img.AddCustomAttributes("src", "http://www.mybligr.com/wp-
content/uploads/2017/03/cute-and-lovable-pictures-of-rabbits-2.jpg");
img.AddCustomAttributes("alt", "rabbit");
img.AddCustomAttributes("height", "300");
img.AddCustomAttributes("width", "400");
img.AddCSSStyle(new
ILib_CSSStyle(ILib_CSSStyle.CSSStyles.border_radius.toString(), "50%"));
stranica.addComponent(img);
return stranica.printPage();
}
}
```

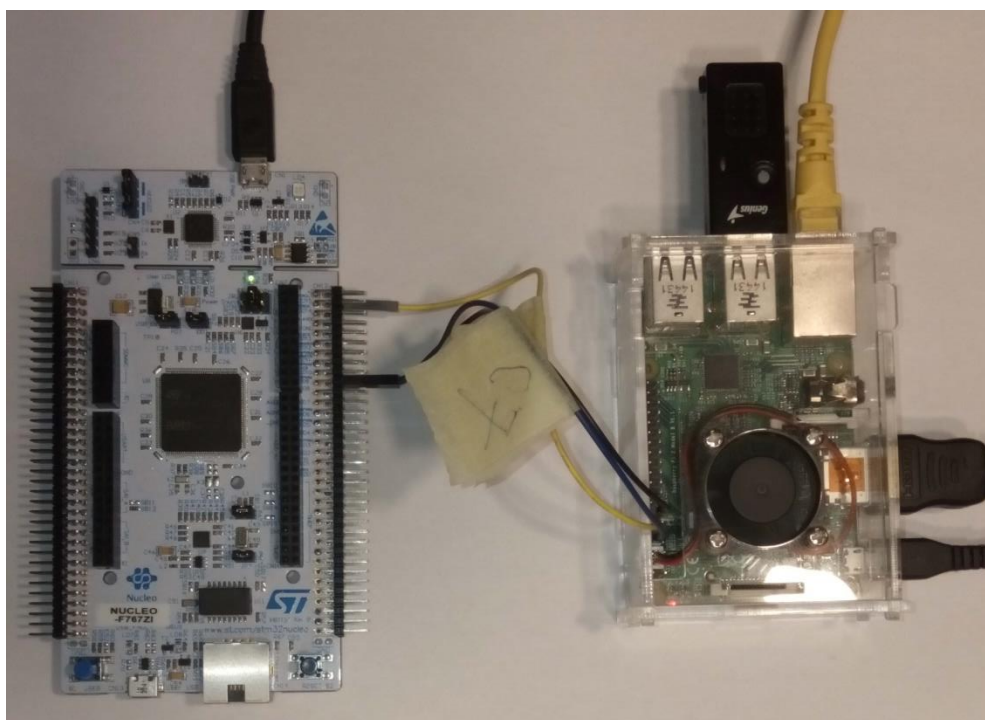
## 6. EMBEDDED ДЕО

Демонстрација се заснива на две компоненте, а то су *software* део и *hardware* део. Пошто смо у прошлим поглављима видели *software* -ске елементе у овом поглављу ћемо се фокусирати на *hardware*.

Сервер се подиже на Raspberry Pi-ју командом:

```
java -jar /home/pi/Desktop/demo-0.0.1-SNAPSHOT.jar
```

На серијском порту `/dev/ttyACM0` је повезан микроконтролер и кад корисник пошаље *GET* захтев са потребним параметрима Raspberry то прослеђује микроконтролеру и онда микроконтролер укључује или искључује одређену линију на порту. На слици 4 се налази уређај.



Слика 4. – Припадни хардвер

```
-----  
-*/  
/* Includes -----  
#include "main.h"  
#include "stm32f7xx_hal.h"  
#include "adc.h"  
#include "usart.h"  
#include "gpio.h"  
  
/* USER CODE BEGIN Includes */  
#include <stdio.h>  
#include <string.h>  
/* USER CODE END Includes */
```

```

-*/      /* Private variables -----
-*/
        /* USER CODE BEGIN PV */
-*/      /* Private variables -----
-*/      #define COUNTOF(__BUFFER__) (sizeof(__BUFFER__) / sizeof(*(__BUFFER__)))
        /* Size of Transmission buffer */
        #define TXBUFFERSIZE (COUNTOF(aTxBuffer) - 1)
        /* Size of Reception buffer */
        #define RXBUFFERSIZE TXBUFFERSIZE

        /* Buffer used for transmission */
        uint8_t aTxBuffer[2] = {'a', 'b'};

        uint8_t da[2] = {'D', 'A'};
        uint8_t ne[2] = {'N', 'E'};

        /* Buffer used for reception */
        uint8_t aRxBuffer[2] = {0x00, 0x00};

        /* USER CODE END PV */

-*/      /* Private function prototypes -----
-*/      void SystemClock_Config(void);

        /* USER CODE BEGIN PFP */
-*/      /* Private function prototypes -----
-*/

        /* USER CODE END PFP */

        /* USER CODE BEGIN 0 */

        /* USER CODE END 0 */

        int main(void)
        {

            /* USER CODE BEGIN 1 */

            /* USER CODE END 1 */

            /* MCU Configuration-----
---*/

            /* Reset of all peripherals, Initializes the Flash interface and the
Systick. */
            HAL_Init();

            /* USER CODE BEGIN Init */

            /* USER CODE END Init */

            /* Configure the system clock */
            SystemClock_Config();

            /* USER CODE BEGIN SysInit */

            /* USER CODE END SysInit */

            /* Initialize all configured peripherals */

```



```
MX_GPIO_Init();
MX_USART6_UART_Init();
MX_ADC1_Init();

/* USER CODE BEGIN 2 */

/* USER CODE END 2 */

/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{
/* USER CODE END WHILE */

/* USER CODE BEGIN 3 */

    HAL_UART_Receive(&huart6, (uint8_t *)aRxBuffer, 2, 2000);

    if(strcmp ("OK", (char*)aRxBuffer) == 0)
    {
        HAL_UART_Transmit(&huart6, (uint8_t *)da, 2, 1000);
        HAL_GPIO_WritePin(GPIOB, GPIO_PIN_7, GPIO_PIN_SET);
    }
    if(strcmp ("NO", (char*)aRxBuffer) == 0)
    {
        HAL_UART_Transmit(&huart6, (uint8_t *)ne, 2, 1000);
        HAL_GPIO_WritePin(GPIOB, GPIO_PIN_7, GPIO_PIN_RESET);
    }

}
/* USER CODE END 3 */

}

/** System Clock Configuration
*/
void SystemClock_Config(void)
{
    RCC_OscInitTypeDef RCC_OscInitStruct;
    RCC_ClkInitTypeDef RCC_ClkInitStruct;
    RCC_PeriphCLKInitTypeDef PeriphClkInitStruct;

    /**Configure the main internal regulator output voltage
    */
    __HAL_RCC_PWR_CLK_ENABLE();

    __HAL_PWR_VOLTAGESCALING_CONFIG(PWR_REGULATOR_VOLTAGE_SCALE1);

    /**Initializes the CPU, AHB and APB busses clocks
    */
    RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSI;
    RCC_OscInitStruct.HSISate = RCC_HSI_ON;
    RCC_OscInitStruct.HSICalibrationValue = 16;
    RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
    RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSI;
    RCC_OscInitStruct.PLL.PLLM = 8;
    RCC_OscInitStruct.PLL.PLLN = 216;
    RCC_OscInitStruct.PLL.PLLP = RCC_PLLP_DIV2;
    RCC_OscInitStruct.PLL.PLLQ = 2;
    if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
    {
        __Error_Handler(__FILE__, __LINE__);
    }
}
```

```
    /**Activate the Over-Drive mode
    */
    if (HAL_PWREx_EnableOverDrive() != HAL_OK)
    {
        _Error_Handler(__FILE__, __LINE__);
    }

    /**Initializes the CPU, AHB and APB busses clocks
    */
    RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
                                   |RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
    RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
    RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
    RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV4;
    RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV2;

    if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_7) != HAL_OK)
    {
        _Error_Handler(__FILE__, __LINE__);
    }

    PeriphClkInitStruct.PeriphClockSelection = RCC_PERIPHCLK_USART6;
    PeriphClkInitStruct.Usart6ClockSelection = RCC_USART6CLKSOURCE_PCLK2;
    if (HAL_RCCEx_PeriphCLKConfig(&PeriphClkInitStruct) != HAL_OK)
    {
        _Error_Handler(__FILE__, __LINE__);
    }

    /**Configure the SysTick interrupt time
    */
    HAL_SYSTICK_Config(HAL_RCC_GetHCLKFreq()/1000);

    /**Configure the SysTick
    */
    HAL_SYSTICK_CLKSourceConfig(SYSTICK_CLKSOURCE_HCLK);

    /* SysTick_IRQn interrupt configuration */
    HAL_NVIC_SetPriority(SysTick_IRQn, 0, 0);
}

void MX_USART6_UART_Init(void)
{
    huart6.Instance = USART6;
    huart6.Init.BaudRate = 9600;
    huart6.Init.WordLength = UART_WORDLENGTH_8B;
    huart6.Init.StopBits = UART_STOPBITS_1;
    huart6.Init.Parity = UART_PARITY_NONE;
    huart6.Init.Mode = UART_MODE_TX_RX;
    huart6.Init.HwFlowCtl = UART_HWCONTROL_NONE;
    huart6.Init.OverSampling = UART_OVERSAMPLING_16;
    huart6.Init.OneBitSampling = UART_ONE_BIT_SAMPLE_DISABLE;
    huart6.AdvancedInit.AdvFeatureInit = UART_ADVFEATURE_NO_INIT;
    if (HAL_UART_Init(&huart6) != HAL_OK)
    {
        _Error_Handler(__FILE__, __LINE__);
    }
}
```

## 7. ЗАКЉУЧАК

На идеју да направим овај пројекат сам дошао док сам радио на једном пројекту. Пошто сам често морао да додајем *HTML* елементе и да их складиштим у *String* променљивима, што је захтевало да стално *escape*-ујем карактере као што су наводници једноструки и двојструки. Друга варијанта је да се ради на истом принципу као што то ради *Angular*, али нисам љубитељ тога пошто нисам примарно фокусиран на веб програмирање. Метод који сам развио у овом пројекту омогућава људима који су примарно *Java* програмери да праве веб странице које могу физички да раде са рачунаром тј. да нису виртуализоване. Апликације изграђене са овом библиотеком омогућавају да се искористе све предности које *Java* пружа *desktop* корисницима. Током развоја овог пројекта приметио сам да оваква апликација може да ради и на полу *embedded* системима као што је *Raspberry Pi*.

**После одбране рада код овог пројекта ће бити објављен под**

**Creative Commons BY-SA лиценцом на github-у и уколико буде интересовања на Maven Repository-ју.**

**This whole project including ILib library is licenced under Creative Commons BY-SA licence.**

## 8. ЛИТЕРАТУРА

- [1][https://sr.wikipedia.org/sr/%D0%88%D0%B0%D0%B2%D0%B0\\_\(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%81%D0%BA%D0%B8\\_%D1%98%D0%B5%D0%B7%D0%B8%D0%BA\)](https://sr.wikipedia.org/sr/%D0%88%D0%B0%D0%B2%D0%B0_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%81%D0%BA%D0%B8_%D1%98%D0%B5%D0%B7%D0%B8%D0%BA))
- [2][https://en.wikipedia.org/wiki/Spring\\_Framework](https://en.wikipedia.org/wiki/Spring_Framework)
- [3][https://sr.wikipedia.org/sr/C\\_\(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%81%D0%BA%D0%B8\\_%D1%98%D0%B5%D0%B7%D0%B8%D0%BA\)](https://sr.wikipedia.org/sr/C_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%81%D0%BA%D0%B8_%D1%98%D0%B5%D0%B7%D0%B8%D0%BA))