

SPOJENÁ ŠKOLA, KOMÁRŇANSKÁ 28, NOVÉ  
ZÁMKY

organizačná zložka

Stredná priemyselná škola elektrotechnická S. A. Jedlika,

S. A. Jedlik Elektrotechnikai Szakközépiskola

GAMESERVER – WEBOVÁ ČASŤ

ZOLTÁN ONÓDY

# **GAMESERVER – WEBOVÁ ČASŤ**

## **KOMPLEXNÁ ODBORNÁ PRÁCA**

**ZOLTÁN ONÓDY**

**Študijný odbor: informačné a sieťové technológie**

**Konzultant: Ing. František Hortai**

**Nové Zámky 2015**

## Čestné vyhlásenie

Čestne vyhlasujem, že prácu som vypracoval sám, bez cudzej pomoci iba s použitím literatúry ktorú som uviedol na poslednej strane dokumentácie.

V Nových Zámkoch, dňa 18.9.2014

---

Zoltán Onódy

## **Pod'akovanie**

Pod'akovať by som chcel hlavne Lawrence Journal-World za vytvorenie tak fantastického frameworku akým je Django, „Maru“ ktorá ma doviedla k tomu aby som sa ho naučil a v neposlednom rade by som sa chcel pod'akovať všetkým profesorom ktorí ma za tie 4 roky učili.

# Obsah

<b>1. ÚVOD.....</b>	<b>4</b>
1.1. ZOZNAM POTREBNÝCH NÁSTROJOV .....	4
1.2. INŠTALÁCIA POTREBNÝCH NÁSTROJOV .....	4
1.3. HTML, CSS A TWITTER BOOTSTRAP .....	5
1.4. PYTHON3.....	6
1.5. DJANGO .....	6
1.6. GIT + BITBUCKET .....	7
<b>2. DIZAJN &amp; ROZLOŽENIE WEB-STRÁNKY .....</b>	<b>8</b>
2.1. NÁVRH ZÁKLADNEJ KOSTRY .....	8
2.2. NAVIGÁCIA A PÄTIČKA .....	8
2.3. ÚVODNÁ STRÁNKA.....	10
2.4. ZOZNAM HIER .....	11
<b>3. VYTVORENIE JADRA WEBU .....</b>	<b>12</b>
3.1. DATABÁZA .....	12
3.2. ADMINISTRÁCIA WEBOVEJ STRÁNKY.....	12
3.3. VYTVÁRANIE SERVERA.....	14
3.4. KONTAKT .....	14
<b>4. RIEŠENIE PROBLÉMOV.....</b>	<b>16</b>
4.1. ZÍSKANIE PRVÉHO VOĽNÉHO PORTU .....	16
4.2. SLUGIFY – KRÁSNE URL .....	17
4.3. ZISTENIE INFORMÁCII O SERVERI .....	18
4.4. VYPÍNANIE NEPOUŽÍVANÝCH SERVEROV.....	18
<b>5. POSLEDNÉ KROKY .....</b>	<b>20</b>
5.1. REGISTRÁCIA DOMÉNY .....	20
5.2. SPOJAZDZENIE WEB-STRÁNKY NA SERVERY.....	20
<b>6. ZÁVER .....</b>	<b>21</b>
<b>7. RESUMÉ .....</b>	<b>22</b>
<b>8. BIBLIOGRAFIA .....</b>	<b>23</b>
<b>9. OBRÁZKOVÁ PRÍLOHA .....</b>	<b>24</b>

# 1. Úvod

Na komplexnú odbornú prácu so si vybral tvorbu webovej stránky pomocou ktorej sa dá ovládať herný server a to z toho dôvodu, že som v tom videl výzvu a zároveň príležitosť ako získať veľa nových poznatkov či už v operačnom systéme Linux, v programovaní v Pythone3 alebo vo vytváraní webových stránok pomocou frameworku Django. V projekte budeme používať tiež CSS framework Twitter Bootstrap, ktorý nám uľahčí prácu na dizajne webstránky, ktorá vďaka tomuto frameworku bude moderná a responzívna. V projekte môžeme naraziť na veľa rôznych problémov, ktoré nás donútia hľadať lepšie riešenia, vďaka ktorým sa z nás stanú lepší webdeveloperi a programári.

## 1.1.Zoznam potrebných nástrojov

Na prácu s frameworkom Django potrebujeme mať nainštalovaných pár ďalších nástrojov ako napríklad interpret pre Python3.4, Pip, Git a virtuálne prostredie. Na úpravu súborov odporúčame použiť IDE ktoré podporuje Django, my budeme používať PyCharm avšak všetko sa dá samozrejme spraviť aj v notepade.

Projekt bude vytvorený pod operačným systémom Ubuntu 14.04LTE. Niektoré kroky sa na rôznych distribúciách Linuxov môžu líšiť.

## 1.2.Inštalácia potrebných nástrojov

Nastal čas aby sme si nainštalovali všetky nástroje ktoré budeme pri tvorbe projektu potrebovať. V prvom rade si stiahneme nástroj Pip3 s príkazom `$sudo apt-get install python3-pip`. Pip3 slúži na jednoduchú inštaláciu nástrojov a balíkov pre jazyk Python3.

Najľahší spôsob ako nainštalovať potrebnú verziu Pythonu je pomocou nástroja Pythonbrew. Pythonbrew je verziovací systém ktorý podporuje ľahkú inštaláciu rôznych verzií Pythonu. Pythonbrew si stiahneme s príkazom `$pip3 install pythonbrew`, Pythonbrew vyžaduje inštaláciu Curl, ten nainštalujeme príkazom `$apt-get install curl`. Pythonbrew je teraz nainštalovaný avšak nefunkčný, príkazom `$pythonbrew install` získame inštrukcie k úspešnému dokončeniu inštalácie, tieto kroky budeme nasledovať. Pythonbrew je teraz plne funkčný tj. môžeme nainštalovať Python3.4.1 pomocou príkazu `$pythonbrew install 3.4.1`.

Novú verziu Pythonu by sme mali mať nainštalovanú, overíme si to pomocou `$pythonbrew list`, teraz túto verziu aktivujeme `$pythonbrew switch 3.4.1` a overíme si či sa nám táto verzia aktivovala príkazom `$python`. Na prvom riadku by sme mali vidieť verziu 3.4.1. Python máme teraz pripravený, prejdeme na inštaláciu virtuálneho prostredia.

Virtuálne prostredie nainštalujeme pomocou Pip3 príkazom `$pip3 install virtualenv`, akonáhle sa nám virtuálne prostredie nainštaluje, môžeme vytvoriť náš prvý projekt pomocou príkazu `$virtualenv web`. Kde „web“ je meno zložky v ktorej budeme mať projekt uložený.

Projekt máme vytvorený, vstúpime doňho pomocou `$cd web`. V projekte budeme Python spúšťať príkazom `$bin/python`. Overíme si, či náš projekt obsahuje najnovšiu verziu Pythonu a následne do projektu nainštalujeme Django pomocou `$bin/pip install django`. Framework Django je teraz nainštalovaný a nám neostáva nič iné ako projekt Django vytvoriť. Projekt vytvoríme pomocou `$bin/django-admin startproject gameserver_project .` Všimnime si bodku na konci príkazu, tá znamená, že nechceme vytvoriť projekt do podadresára gameserver\_project ale do adresára v ktorom sa práve nachádzame. Projekt máme vytvorený a pomocou príkazu `$bin/python manage.py runserver` ho spustíme. Projekt sa nám spustil a keď prejdeme na adresu ktorú vidíme v termináli (pravdepodobne 127.0.0.1:8000) tak by sme mali vidieť náš projekt spustený.



Obrázok 1 – Ukážka funkčného projektu

### 1.3.HTML, CSS a Twitter Bootstrap

HTML – Hypertextový značkovací jazyk, je určený na tvorbu webových stránok. Pomocou tohoto jazyka sa na webovú stránku dajú pridať odkazy, obrázky, nadpisy, paragrafy atď. Bol vytvorený v roku 1993 firmami W3C a WHATWG, voľakedy sa pomocou neho robil aj dizajn stránok, dneska sa používa len na tvorbu obsahu.

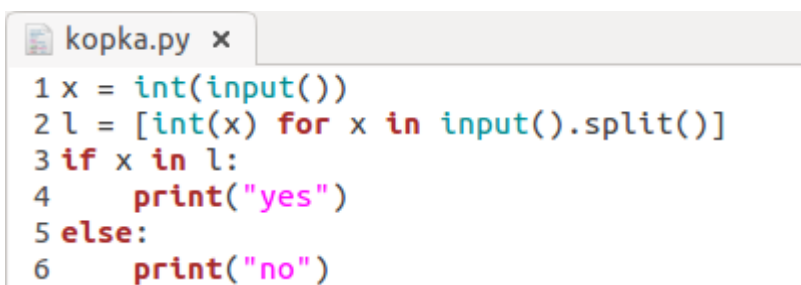
CSS – Kaskádové štýly, sú určené na tvorbu dizajnu webových stránok. Vďaka tomuto jazyku vyzerajú webové stránky v súčasnosti tak dobre. Jazyk CSS bol vytvorený v roku 1996 firmou W3C. Aktuálna verzia je CSS3 ktorá však stále nemá plnú podporu v prehliadačoch ako napríklad Internet Explorer.

Twitter Bootstrap – je webový front-end framework napísaný v HTML, CSS, LESS, SASS a v JavaScripte. Je to vlastne kolekcia pred-pripravených štýlov, komponentov a skriptov. Bol vytvorený firmou Twitter pre ich vlastné použitie, avšak v Auguste 2011 tento framework vydali ako Open-Source. Vďaka tomuto frameworku vieme vytvoriť moderný, responsívny web za pár minút.

### 1.4.Python3

Python je rozšírený, multiplatformový, programovací jazyk s veľmi prehľadným syntaxom, vďaka čomu je tento jazyk vhodný pre aj pre začiatočníkov. Python bol navrhnutý tak aby sa na pár riadkov dali napísať pomerne komplexné programy. Rovnaký program napísaný v Pythone má priemerne 5-10x menej riadkov ako program napísaný v C++. Python používajú back-end programátori webových stránok, softvéroví vývojári, vedci a systémoví administrátori na automatizáciu. Python vytvoril Guido van Rossum v roku 1991. V Pythone je napísaných mnoho programov aj webových stránok ako napríklad Yahoo Maps, Reddit či Youtube.

Ukážka programu v Python3. Na prvý riadok vstupu zadáme číslo ktoré chceme zistiť či sa nachádza v zozname a na druhý riadok zadáme zoznam medzerami oddelených celých čísiel. Program v Python3 má 6riadkov a 106znakov, rovnaký program v C++ má 31 riadkov a 372znakov.



```
kopka.py x
1 x = int(input())
2 l = [int(x) for x in input().split()]
3 if x in l:
4     print("yes")
5 else:
6     print("no")
```

Obrázok 2 – Ukážka programu v Python3

### 1.5.Django

Django je webový back-end framework napísaný v Pythone ktorý zrýchľuje tvorbu webstránok a podporuje čistý, prehľadný kód. Django používa architektúru MVC – Model View Controller ktorá oddeluje databázovú, prezentačnú a logickú časť kódu. Framework Django vytvorila firma Lawrence Journal-World v roku 2005. Aktuálnu verzia Djanga (1.7) vydali začiatkom roka 2014. Django používajú aj veľké webstránky ako playfire.com, disqus.com a washingtonpost.com.



## 1.6.GIT + Bitbucket

Git je distribuovacia systém ktorý vytvoril Linus Torvalds. Slúži na správu revízií, my ho v projekte budeme používať na zálohovanie. Samotný GIT je len softvér tj. neposkytuje priestor v ktorom by sme mohli mať jednotlivé revízie uložené. Server na ktorom môžu byť tieto revízie uložené by sme si mohli vytvoriť aj sami ale to nie je cieľom tejto práce. Najznámejší server ktorý poskytuje takéto služby je GitHub avšak vo free balíčku nie je možné mať projekt skrytý pre verejnosť. Kvôli tomu som sa rozhodol pre menej známy server Bitbucket.

Bitbucket ako som už spomenul je služba ktorá poskytuje priestor na zálohovanie kódu pomocou softvéru GIT. Na webovej stránke Bitbucketu sa najprv zaregistrujeme, následne dáme vytvoriť novú schránku, zaškrtneme aby bola schránka súkromná a schránku vytvoríme. Po vytvorení sa nám zobrazia 2 kroky, prvý pre prípad, že projekt ešte nemáme vytvorený a druhý už pre existujúce projekty. Nás zaujíma tá druhá možnosť.

Spojzadenie GITu je veľmi ľahké, stačí použiť nasledovné príkazy:

```
$cd /cesta/k/projektu;
```

```
$git init;
```

```
$git remote add origin https://meno@bitbucket.org/meno/projekt.git;
```

```
$git push -u origin --all;
```

Na používanie GITu potrebuje poznať ešte 3 príkazy: `$git add --all` slúži na vytvorenie revízie, `$git commit -m "popis zmien"` slúži na popis danej revízie, toto je dôležité najmä do budúcnosti aby sme mali prehľad o tom čo sa v ktorej verzii projektu pridalo, `$git push` slúži na odoslanie revízie na server.

## 2. Dizajn & Rozloženie web-stránky

### 2.1.Návrh základnej kostry

Základná kostra web-stránky musí obsahovať nejaké povinné prvky, hlavičku ktorá obsahuje skripty a telo ktoré obsahuje obsah stránky. Do hlavičky sa píše veci ako kódovanie, odkazy na kaskádové štýly a skripty, meno karty atď. Zo stránky [getbootstrap.com](http://getbootstrap.com) si stiahneme potrebné kaskádové štýly a skripty, následne ich pridáme do základnej kostry webovej stránky. V konečnom dôsledku základná kostra vyzerá nasledovne.

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="utf-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1">
7   <title>Domov | V-Servers.eu</title>
8   <!-- Bootstrap core -->
9   <link href="/static/css/bootstrap.min.css" rel="stylesheet">
10  <link href="/static/css/starter-template.css" rel="stylesheet">
11 </head>
12 <body>
13   <!-- obsah stránky pôjde sem -->
14
15   <!-- Bootstrap core JavaScript -->
16   <script src="/static/js/jquery.js"></script>
17   <script src="/static/js/bootstrap.min.js"></script>
18 </body>
19 </html>
```

- 1) Pomocou doctype dáme prehliadaču vedieť akú verziu HTML budeme používať, v našom prípade HTML5.
- 4) Charset, nastaví kódovanie znakov na webovej stránke, ak by sme žiadny charset nenastavili, diakritika by sa nám nezobrazovala správne.
- 7) Text ktorý sa nachádza v title sa nám bude zobrazovať na karte v prehliadači.
- 9) Link slúži na pripojenie CSS súborov k webovej stránke.
- 16) Script funguje rovnako ako link s tým rozdielom že pripája skripty.

### 2.2.Navigácia a pätička

Ako som už spomenul, vytvorenie moderného, responzívneho a celkom pekného dizajnu vďaka knižnici Twitter Bootstrap zaberie len chvíľu. Stačí nám napísať pár riadkov HTML kódu v ktorom použijeme už vopred pripravené triedy.

```

1 <div class="navbar navbar-inverse navbar-fixed-top" role="navigation">
2   <div class="container">
3     <div class="navbar-header">
4       <button type="button" class="navbar-toggle collapsed"
5         data-toggle="collapse" data-target=".navbar-collapse">
6         <span class="sr-only">Toggle navigation</span>
7         <span class="icon-bar"></span>
8         <span class="icon-bar"></span>
9         <span class="icon-bar"></span>
10      </button>
11      <a class="navbar-brand" href="/">V-Servers.eu</a>
12    </div>
13    <div class="collapse navbar-collapse">
14      <ul class="nav navbar-nav">
15        <li><a href="/create/">Vytvorit server</a></li>
16        <li><a href="/about/">O nás</a></li>
17        <li><a href="/blog/">Blog</a></li>
18      </ul>
19    </div><!--/.nav-collapse -->
20  </div>
21 </div>

```

Týchto 21 riadkov „kódu“ nám vytvorí celkom pekné menu ktoré sa prispôbi veľkosti displeja. Nás zaujíma hlavne navbar-nav kam pridáme odkazy ktoré chceme mať na našej webovej stránke. Pre bližšie informácie o jednotlivých prvkoch v menu navštívte dokumentáciu Twitter Bootstrap.

```

1 <hr>
2 <div class="row">
3   <div class="col-md-12">
4     <p class="text-center">
5       &copy; 2014 - All right reserved - just kidding, no rights reserved
6     </p>
7   </div>
8   <div class="col-md-12">
9     <p class="text-center">
10      <a href="/about/"></a>
11    </p>
12  </div>
13 </div>

```

1) hr nám pridá vodorovnú čiaru na webovú stránku ktorá bude oddelovať obsah webovej stránky od pätičky

4) knižnica Twitter Bootstrap nám okrem iného ponúka aj triedu text-center ktorá nám zarovnáva text do stredu webovej stránky

Pre „pozretie“ základnej kostry webovej stránky ktorú zatiaľ tvorí menu a pätička, pozrite v prílohe obrázok [1.1] pre normálne zobrazenie a obrázok [1.2] pre mobilné zobrazenie.

## 2.3.Úvodná stránka

Keďže webová stránka nie je zameraná na obsah ale funkčnosť a dizajn, jediné čo na úvodnej stránke budeme mať je takzvaný „carousel“ resp. „slideshow“ v ktorom budeme zobrazovať naše najžiadanejšie hry.

```
1 <div id="carousel-example-generic" class="carousel slide" data-ride="carousel">
2   <ol class="carousel-indicators">
3     <li data-target="#carousel-example-generic" data-slide-to="0" class="active"></li>
4     <li data-target="#carousel-example-generic" data-slide-to="1"></li>
5   </ol>
6   <div class="carousel-inner">
7     <div class="item active">
8       
9       <div class="carousel-caption"><h2>Call of Duty 4: Modern Warfare</h2></div>
10    </div>
11    <div class="item">
12      
13      <div class="carousel-caption"><h2>Counter Strike - Global Offensive</h2></div>
14    </div>
15  </div> <!-- Controls -->
16  <a class="left carousel-control" href="#carousel-example-generic" role="button" data-slide="prev">
17    <span class="glyphicon glyphicon-chevron-left"></span>
18  </a>
19  <a class="right carousel-control" href="#carousel-example-generic" role="button" data-slide="next">
20    <span class="glyphicon glyphicon-chevron-right"></span>
21  </a>
22 </div>
```

1) Všimnime si volanie 2 tried a to triedy carousel a slide, tieto 2 triedy obsahujú vopred našťýlované prvky vďaka ktorým carousel nemusíme našťýlovať my.

2) Indikátory sú tie malé krúžky na spodku carousela pomocou ktorých sa dá priamo prepínať medzi jednotlivými obrázkami.

3) Počet guľičiek musí byť rovný počtu obrázkov ktoré máme v carousel, data-slide-to indikuje na ktorý obrázok sa má carousel po kliknutí presunúť. Tak ako v programovaní aj tu číslovanie začína od nuly. Všimnime si, že prvý indikátor má triedu active, vďaka tomu je hneď po načítaní zobrazená, ak by sme triedu active nemali ani pri jednom indikátory, carousel by sa nám zobrazoval prázdny počas prvých 4 sekúnd kým by sa neaktivoval iný prvok.

6) Tento element má triedu carousel-inner ktorým vlastne povieme prehliadaču, že synovia tohoto elementu budú prvky ktoré sa majú na stránke zobrazovať.

7) Elementy s triedou item sa považujú za prvky ktoré sa budú postupne zobrazovať. Jeden z týchto prvok však musí mať aj triedu active už z vyššie spomínaných dôvodov. Tento element musí obsahovať obrázok, ten sa vykreslí na celú výšku a šírku a môže obsahovať aj text ktorý bude vykreslený na obrázku.

16) Poslednou dôležitou časťou tohto carousela je takzvaný carousel-control vďaka ktorému sa dá pohybovať medzi jednotlivými obrázkami pomocou šípok na boku.

Pre náhľad carousela si pozrite obrázok [2] v prílohe.

## 2.4.Zoznam hier

Zoznam hier je stránka na ktorej je „nečakane“ zoznam hier ktoré na našom servery poskytujeme, je jednoduchá, v skutočnosti je to len pár obrázkov hier cez ktoré sa dá prekliknúť ku konfigurácii jednotlivých serverov.

```
1 <div class="col-lg-8 col-lg-offset-2 col-">
2   <div class="col-md-6">
3     <a href="/create/5/" class="thumbnail">
4       
5     </a>
6   </div>
7   <div class="col-md-6">
8     <a href="/create/6/" class="thumbnail">
9       
10    </a>
11  </div> <!-- atd ... -->
12 </div>
```

- 1) Toto je hlavný element stránky, keďže hier je málo a obrázky nie sú kvôli dizajnu veľké, stránku musíme rozdeliť na 8/12 z celkovej šírky, o to sa stará trieda col-lg-8. Element síce už má veľkosť akú potrebujeme avšak je zarovnaný v ľavo. Najľahší spôsob ako ho zarovnať do stredu je odsadiť ho o 2/12 z celkovej šírky z ľavej strany pomocou triedy col-lg-offset-2.
- 2) Okrem toho, že rozdelíme hlavný element musíme rozdeliť aj tieto podelementy ktoré budú vlastne pokrývať jednotlivé obrázky tentokrát však na 1/2 to pomocou triedy col-md-6.
- 3) Posledným dôležitým prvom je triedy thumbnail ktorá nám naštýluje rámček okolo obrázkov.

Pre náhľad zoznamu hier si pozrite obrázok [3] v prílohe.

### 3. Vytvorenie jadra webu

#### 3.1.Databáza

Najdôležitejšie pri práci s databázami je ich návrh, keďže už bežiacu databázu sa dá iba ťažko upravovať. Django nám tieto dodatočné úpravy umožňuje, ale to nič nemení na fakte, že by sme to nemali robiť. Naša databáza sa bude skladať z 2 hlavných častí a to časť Gameserver a časť Blog.

Gamserver je dosť komplexná databáza na pomery takýchto projektov. Má tabuľky zvlášť pre vytvorený server, počet hráčov, mapy, typy hier, hry a módy. V tabuľke hry máme uložené informácie ako meno hry, špeciálny názov s ktorým pracujeme v kóde, cesty k skriptom a k adresárom, popis, adresu obrázka atď. V tabuľke server si pamätáme konfiguráciu daného servera tj. hru, počet hráčov, typ hry, port, heslá, hodnotu servera ktorá slúži na jeho vypínanie atď.

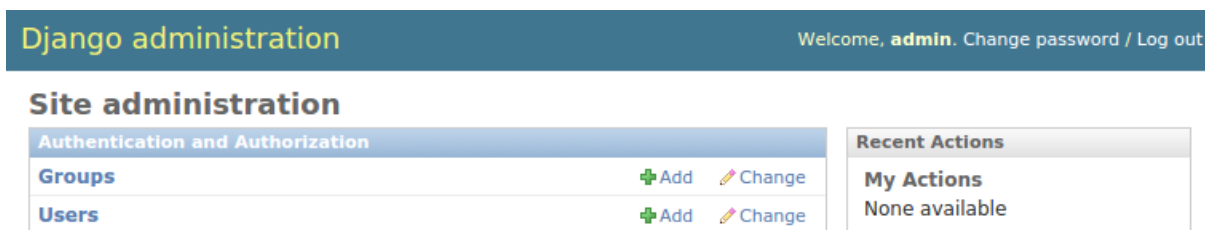
Blog sa skladá z 3 častí a to kategórie, značky a články. Kategórie a značky obsahujú 3 polia, prvé pre meno, druhé pre popis a tretie pre slug. Články sa skladajú z nadpisu, booleanovského pola ktorý indikuje či má byť článok zobrazený, contextu ktorý slúži na krátky popis obsahu, slug, povolenie komentárov a samozrejme značky a kategórie.

Kódy k jednotlivým databázam sa nachádzajú v súboroch models.py.

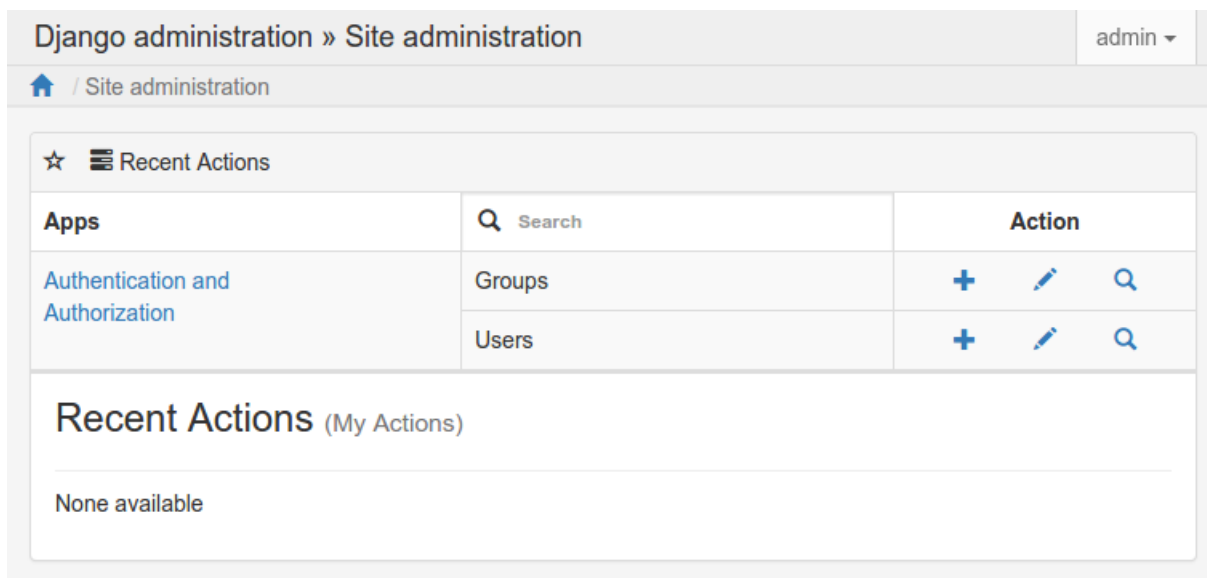
#### 3.2.Administrácia webovej stránky

Samotné Django ponúka ľahkú administráciu webových stránok a už pri vytvorení prázdneho projektu sa nám vytvorí administráciu ktorú nájdeme na adrese /admin/. Ak prejdeme do administrácie zistíme, že nemáme prihlasovacie údaje, Django totiž nevytvorí žiadny administratívny účet, ten si musíme vytvoriť ručne pomocou terminálu. Použijeme príkaz `$bin/python manage.py createsuperuser`, po stlačení klávesy enter, si od nás terminál vyžiada meno, email a heslo. Po vyplnení údajov je administratívny účet plne funkčný tj. už sa bez problémov vieme dostať do administrácie.

V administrácii Django toho však veľa zatiaľ nenájdeme, je tam len pokročilá správa užívateľov a skupín. Administrácia stránky má však celkom starý dizajn, ktorý bohužiaľ nespĺňa naše požiadavky.



Ostávajú nám teda 3 možnosti, 1. uspokojiť sa s tým čo máme, 2. preprogramovať administráciu Django, čím by sme si ju prispôbili podľa našich predstáv to by nám však zabralo desiatky hodín práce. 3. pohľadať na internete nejaký krajší template a ten nainštalovať. My sme sa rozhodli pre 3. možnosť a tak sme našli tému `bootstrap_admin` ktorá je moderná a responzívna. Nainštalujeme si ju pomocou `$bin/pip install bootstrap_admin` a aktivujeme si tak, že ju pridáme do zoznamu aktívnych aplikácií v `gameserver_project/settings.py`. Ak sa nám nezobrazuje správne, tak to z toho dôvodu, že sa nám do Django nestiahli potrebné štýly a skripty, to docielime príkazom `$bin/python manage.py collectstatic`.



Ako vidíme už máme modernú a responzívnu administráciu ktorá však ešte neobsahuje nič užitočné. Na to aby sa nám v administrácii zobrazili informácie o blogu a herných serveroch, musíme tieto veci najprv zaregistrovať. Ukážeme si teda ako sa registruje napríklad taký blog.

Otvoríme si súbor `blog/admin.py` a vytvoríme si triedu ktorá bude dediť z triedy `ModelAdmin`, do tejto triedy vložíme `list_display`. Veci ktoré napíšeme do `list_display` sa nám budú zobrazovať v administrácii pri všeobecnom náhľade na zoznam článkov. Nastavíme aj automatické vytváranie slug-ov vid' odstavec riešenie problémov. Ďalej si nastavíme `fieldsets` čím vlastne zoskupíme jednotlivé polia kvôli prehľadnosti. Triedu zaregistrujeme do administrácie a všetko funguje tak ako má.

```
1 class ArticleAdmin(admin.ModelAdmin):
2     list_display = ['headline', 'category', 'date', 'active', 'comments']
3     prepopulated_fields = {'url': ('headline',)}
4     fieldsets = (
5         (None, {'fields': ('headline', 'url')}),
6         ('Article: ', {'fields': ('context', 'content', 'category', 'tags')}),
7         ('Advanced settings: ', {'fields': ('active', 'comments')}),
8     )
9
10 admin.site.register(Article, ArticleAdmin)
```

### 3.3. Vytváranie servera

Pri vytváraní servera si užívateľ nakliká konfiguráciu z možností ktoré mu nastavíme v administrácii. Väčšinou má na výber zo zoznamu možností ale niektoré polia ako napríklad názov servera a heslo si musí napísať sám. Pre menej skúsených užívateľov je vedľa niektorých polí vysvetlenie na čo slúžia aby ich vedel vyplniť. Po odoslaní konfigurácie na server sa požiadavka začne spracovávať, v prvom rade sa overí či sú všetky povinné polia vyplnené, pokiaľ nie sú tak užívateľovi zobrazí chybová hláška.

**Call of Duty 4**

Názov servera	<input type="text" value="Zoli's server"/>	<input data-bbox="970 689 1010 734" type="button" value="?"/>
Počet hráčov	<input type="text" value="24"/>	
Rcon heslo	<input type="text"/>	<input data-bbox="970 824 1010 869" type="button" value="?"/>
	<div>Toto pole je povinné.</div>	<div>Čo to znamená? Heslo na server s ktorý môžete meniť nastavenia.</div>
Heslo na server	<input type="text"/>	
Mód	<input type="text" value="pml220"/>	
Typ hry	<input type="text" value="Search &amp; Destroy"/>	
Promod mód	<input type="text" value="match_mr12"/>	
Map	<input type="text" value="Crossfire"/>	
<input type="button" value="Vytvoriť"/>		

Ak je formulár validný tj. užívateľ nakonfiguroval server správne, tak sa požiadavka spracuje. V prvom rade si zistíme prvý voľný port (viď. odsek riešenie problémov) pokiaľ nie je žiaden voľný port k dispozícii užívateľovi sa zobrazí chybová hláška. Kvôli bezpečnosti si musíme vygenerovať heslo, vďaka ktorému budeme vedieť pristupovať ku konfigurácii servera cez jedinečnú adresu. Heslo má kvôli bezpečnosti až  $10^{15}$  kombinácií, prerazenie tohto hesla takzvanou hrubou silou by terajším počítačom trvalo rádovo až 120 dní. Následne sa vytvoria potrebné súbory pre vytvorenie hry ako spúšťačí skript a konfigurácia servera. Teraz si už len skript spustíme a pošleme mu parameter start. Server sa nám vytvorí a stránka nás presmeruje na stránku s informáciami o serveri ktoré sa získavajú v reálnom čase.

### 3.4. Kontakt

Kontaktný formulár sa nachádza na stránke „O nás“, má 3 hlavné časti, predmet, email a späva. Vytvorenie takéhoto formulára je v Django celkom jednoduché. Stačí, ak vytvoríme vlastnú triedu pre formulár ktorá zdedí vlastnosti triedy Form ktorú ponúka Django. Pomocou




takzvaného widgetu si formulár nakonfigurujeme podľa našich potrieb, nastavíme mu napríklad atribúty CSS triedy vďaka ktorým nadobudne krásny vzhľad. Náš kód vyzerá nasledovne.

```
1 class ContactForm(Form):
2     subject = CharField(widget=TextInput(attrs={'class': 'form-control'}), label="Predmet")
3     sender = EmailField(widget=EmailInput(attrs={'class': 'form-control'}), label="Váš email")
4     message = CharField(widget=Textarea(attrs={'class': 'form-control'}), label="Vaša správa")
```

Overovanie formulára prebieha nasledovne, v prvom rade sa overí či bol formulár vôbec odoslaný, pokiaľ nie tak sa formulár na stránke zobrazí. Ak už bol odoslaný tak sa skontroluje či sú všetky povinné polia vyplnené, ak áno tak sa skúsi odoslať email, ak by náhodou došlo k chybe na serveri tak sa chyba odchyť a užívateľ bude o nej informovaný. Pokiaľ sa žiadna chyba nevyskytne a formulár sa úspešne odošle a užívateľ bude o tom informovaný.

```
1 if request.method == 'POST':
2     form = ContactForm(request.POST)
3     if form.is_valid():
4         try:
5             send_mail(request.POST['subject'], request.POST['message'], request.POST['sender'], ['test@gmail.com'])
6             messages.add_message(request, messages.SUCCESS, 'Email bol odoslaný', extra_tags='alert-success')
7         except:
8             messages.add_message(request, messages.ERROR, 'Email nebol odoslaný', extra_tags='alert-danger')
9         return HttpResponseRedirect('/about/')
10 else:
11     form = ContactForm()
```

Formulár je teraz plne funkčný avšak nemáme žiadnu ochranu proti cudzím skriptom. Takto by sa nám mohlo stať, že by nás niekto zaspamoval. Najjednoduchší spôsob ako tomu predísť je použiť Captchu. Django však žiadnu neponúka čiže si ju musíme doinštalovať. Najlepšia pre naše potreby je django-simple-captcha. Nainštalujeme si ju pomocou príkazu `$bin/pip install django-simple-captcha`, v nastaveniach projektu tj. `gameserver_project/settings.py` si túto aplikáciu pridáme do zoznamu nainštalovaných aplikácií. Captcha je pripravená na používanie, pridáme si ju teda do našej triedy, ktorú sme si popísali o 2 odstavce vyššie.

<b>Predmet</b>	<input type="text" value="Spolupráca"/>
<b>Váš email</b>	<input type="text" value="zoltan.onody@gmail.com"/>
<b>Vaša správa</b>	<div><p>Dobrý deň, mal by som záujem spolupracovať na tomto zaujímavom projekte, mám veľké skúsenosti s programovaním Django aplikácií. Pokiaľ by ste mali záujem o spoluprácu neváhajte ma kontaktovať a dohodneme sa na detailoch.</p><p>S pozdravom, Zoltán Onódy</p></div>
<b>Captcha</b>	<div> <input type="text" value="RNXW"/></div>
<input type="button" value="Odoslať"/>	

## 4. Riešenie problémov

### 4.1. Získanie prvého voľného portu

**Problém:** Keďže na serveri je obmedzený počet portov, treba zistiť ktorý port je voľný a priradiť ho hernému serveru. Na vstupe máme  $N$  = počet portov ktoré môžu byť využité a pole  $L$  = hodnoty využívaných portov, veľkosť poľa  $L$  budeme označovať  $Q$ .

**Riešenie 1:** Zistiť prvý voľný port na serveri vieme hrubou silou, tkzv. „Brute Force“ pre každé číslo od 1 po  $N$  zistíme, či sa nachádza v poly používaných čísiel.

```
1 def get_first_port(l, n):
2     for i in range(n):
3         if not i + 1 in l:
4             return i + 1
5     return -1
```

Problém tohto riešenia je jeho časová zložitosť  $O(Q \cdot N)$ , pre veľký počet prvkov by zbytočne spomaľoval server. Napríklad pre  $n = 1000$  by server musel vykonať rádovo 1000 000 operácií.

**Riešenie 2:** Každého lepšieho programátora (nie web-developera) napadne, že pole si najprv utriedime a potom vyhladáme prvý voľný port.

```
1 def get_first_port(l, n):
2     for i in range(n):
3         if not i + 1 in l:
4             return i + 1
5     return -1
```

Toto riešenie je už dobré, jeho časová zložitosť je  $O(Q + N \log N)$  pre  $n = 1000$  by vykonalo rádovo 11000 krokov. Pomocou binárneho vyhľadávania by sa časová zložitosť dala stiahnuť na  $O(\log Q + N \log N)$  pre  $n = 1000$  by to znamenalo rádovo 10020 krokov, táto optimalizácia však nestojí za čas premrhaný písaním kódu. Otázkou je, existuje rýchlejšie riešenie?

**Riešenie 3:** Každého dobrého programátora napadne vytvoriť booleanové pole veľkosti  $N$ , ktoré je inicializované na False. Pre každý použitý prvok zmeníme pozíciu v pomocnom poli na True. Neskôr si pomocou for cyklu overíme voľné pozície, prvá pozícia v poli ktorá bude mať hodnotu False je zároveň prvý voľný port.

```

1 def get_first_port(l, n):
2     arr = [False for _ in range(n+1)]
3
4     for x in l:
5         arr[x] = True
6
7     for i, x in enumerate(arr[1::]):
8         if not x:
9             return i+1
10    return -1

```

Časová zložitosť tohto riešenia je  $O(N)$ , pre  $N = 1000$  to rádovo vykoná 1000 krokov.

## 4.2.Slugify – krásne URL

Vytvoriť blog v Django ak vieme čo robíme trvá hodinu, avšak keď nepoznáme framework dostatočne narazíme na množstvo problémov ktoré sa riešia celkom pracne. Jedným z nich sú aj krásne URL tj. url v tvare /kategoria/nadpis-clanku/.

**Problém:** Máme nadpis, pre ktorý treba vytvoriť krásne URL.

**Riešenie 1:** Vytvoríme si funkciu ktorá z nadpisu odstráni diakritiku, špeciálne znaky, medzery nahradí pomlčkami atď. Výhodou tohto riešenia je to, že sa pri ňom môžeme naučiť nové programátorské techniky. Nevýhodou je však všetko ostatné, stratený čas, nefunkčnosť riešenia v špeciálnych prípadoch a nečitateľnosť kódu.

**Riešenie 2:** Keďže tento problém je dosť všeobecný pravdepodobne na internete už existuje jeho riešenie. Po chvíli hľadania nájdeme v dokumentácii Djanga vstavanú funkciu ktorá robí presne to čo chceme, tj. z textu ktorý mu vložíme nám vráti text ktorý sa skladá len znakov ktoré môžu byť URL adrese. V zápätí túto funkciu môžeme implementovať do nášho kódu. Namiesto 200 riadkov nám teda stačilo napísať 6.

```

1 def save_model(self, request, obj, form, change):
2     if obj.url is not '':
3         obj.url = slugify(obj.url)
4     else:
5         obj.url = slugify(obj.headline)
6     obj.save()

```

Avšak čo sa stane keď budú mať 2 články rovnaký nadpis? Stránka nám padne keďže nebude vedieť ktorú z nich chceme skutočne zobraziť.

**Riešenie 3:** V models.py nastavíme, aby stĺpec slug bol unikátny, tým predídeme problémom s 2 rovnakými nadpismi. Ďalším vylepšením je, že namiesto toho aby sme v administrácii pri stlačení buttonu na odoslanie odchytili údaje a dopísali slug ako v 2.

riešení, použijeme vstavanú funkciu Django. Nielenže nám to ušetrí 5riadkov kódu ale dokonca nám Django bude slug vykresľovať v reálnom čase, tj. okamžite budeme vidieť ako bude slug vyzerat', na rozdiel od predchádzajúceho riešenia kde sa nám slug objavil až po odoslaní formulára.

```
1 prepopulated_fields = {'slug': ('title',)}
```

### 4.3.Zistenie informácií o serveri

**Problém:** Pri vytvorení servera by sa hodilo ak by sme o tom serveri mali aj nejaké informácie zobrazené priamo na webovej stránke.

**Riešenie:** Pre hry od Valve existuje knižnica vďaka ktorej sa dá tieto informácie získať ľahko. Pre ostatné hry napríklad Call of Duty 4 treba napísať vlastný skript, alebo použiť skript ktorý vytvoril niekto iný. Keďže napísať vlastný skript by trvalo zbytočne dlho (učiť sa API hry atď.) a v programovaní je pravidlo „Do not reinvent the wheel“ my sme použili už hotový skript.

**Problém 2:** Keďže rôzne hry majú rôzne konfigurácie, vracajú rôzne informácie o serveri, to je však zlé, keďže hry potom nevieme obsluhovať všeobecne ale potrebujeme pre každú hru vytvoriť samostatný view, controller, template atď.

**Riešenie 2:** Namiesto toho aby sme všetky hry obsluhovali samostatne je lepší nápad ak si vytvoríme skript, ktorý formát informácií o hre Call of Duty 4 zmení na formát aký majú hry od Valve. Týmto sme docielili to, že nám stačí mať jeden view, jeden controller, jednu administráciu atď.

### 4.4.Vypínanie nepoužívaných serverov

**Problém:** Napriek tomu, že ľudia by mali po skončení hrania server vypnúť veľa z nich stránku zavrie a server beží ďalej. Toto je veľký problém z dvoch dôvodov, po prvé porty sa na servery neuvolní a tým pádom sa po určitom čase minú, po druhé sa zbytočne znižuje výkon servera.

**Riešenie:** Každých X minút kontrolovať či je na serveri niekto pripojený a pokiaľ nie, zapísať si to. Pokiaľ na serveri nie je nikto 3krát za sebou, tak sa server považuje za neaktívny a môže sa vypnúť.

**Implementácia:** Každých X minút server zavolá funkciu shut\_down() pomocou Cronu. Pre každý vytvorený server v našej databáze zistíme počet hráčov ktorý sa na ňom nachádzajú, ak je rovný 0 tak sa nám v databáze pre daný server odráta hodnota 1, ak sa na serveri nachádza aspoň jeden hráč tak sa nám hodnota naopak navýši avšak platí podmienka

hodnota = max(hodnota, 3) tým pádom napriek tomu, že sa na serveri bude hrať 5 hodín kuse, ak hráči odídu zo servera tak sa server vypne rovnako po rádovo 15 minútach rovnako ako by na serveri hrali len hodinu.

```
1 def shut_down():
2     servers = Server.objects.all()
3
4     for server in servers:
5
6         server_info = get_server_info(server.game.special_name, server.port)
7         if server_info == -1 or server_info['player_count'][1][0] == '0':
8             server.value -= 1
9         else:
10            server.value += 1
11            server.value = min(server.value, 3)
12
13        server.save()
14
15        if server.value == 0:
16            system(server.path_to_script + ' stop')
17            server.delete()
```

## 5. Posledné kroky

### 5.1.Registrácia domény

Na Slovensku je veľký počet registrátorov domén, najznámejší je však websupport.sk, jeho ceny však nie sú určené pre študentov, oveľa prijateľnejšie ceny má wedos.cz, kde si doménu kúpime aj my.

Po zakúpení domény, AAA záznamy smerujú pravdepodobne na servery poskytovateľa, tie však potrebujeme zmeniť, preto sa prihlásime do administrácie Wedosu. V menu si nájdeme kolónku domény, vyberieme si našu doménu, klikneme na editovať DNS záznamy. Nás zaujímajú A záznamy a AAA záznamy v prípade, že náš server poskytuje IPv6 adresu. A záznam upravíme, názov necháme prázdny čo znamená, že editujeme záznam pre hlavnú doménu a nie pre subdoménu, TTL necháme prednastavených 1800 a do kolónky dáta vložíme IP adresu nášho servera. Formulár uložíme. Zmeny sa prejavia maximálne do 24 hodín, ale zvyčajne to trvá 10 minút.

### 5.2.Spojzdenie web-stránky na serveri

Webová stránka je vytvorená, doména je zakúpená, jediné čo nám ostáva je nahodiť webovú stránku na server. Keďže poskytovateľ servera ponúka FTP, bežný web by bolo jednoduché nahodiť, avšak ten náš pracuje s dátami, s procesmi a so skriptami takže mu musíme nastaviť rôzne oprávnenia. Na server sa preto prihlásime pomocou terminálu a SSH. Na prihlásenie pomocou SSH použijeme (nečakane) `$ssh root@v-servers.eu` následne si od nás server vypýta heslo. Po prihlásení na server sa presunieme do adresára `$cd /var/www` kde vytvoríme adresár `$mkdir gameserver` a vytvoríme virtuálne prostredie `$virtualenv .` . Teraz je už všetko pripravené na nahodenie webovej stránky, tú stiahneme pomocou gitu príkazom `$git clone cesta_k_projektu@bitbucket.com` . Všetky dáta sú už na serveri avšak web je ešte nefunkčný, kvôli tomu, že Apache ešte nezaregistroval zmeny. Apache preto reštartujeme pomocou `$service apache2 restart`. Ak otvoríme prehliadač zistíme, že webová stránka sa zobrazuje, web funguje avšak opak je pravdou. Akonáhle skúsime vytvoriť server, stránka nám padne s chybovou hláškou „permission denied“ a to z toho dôvodu, že majiteľ adresára je root namiesto www-data. Posledný krok je teda zmena vlastníka priečinka `$chown -R 33:33 /var/www/gameserver` týmto príkazom sme zmenili majiteľa priečinka. Opäť otvoríme prehliadač, prejdeme na našu webovú stránku a zistíme, že všetko funguje ako má. Good job!

## **6. Záver**

V tomto projekte sme sa naučili pracovať s webframeworkom Django, vytvorili sme si Blog ktorý má základnú funkcionality a vytvorili sme samozrejme aj Gameserver vďaka ktorému je možné konfigurovať a vytvárať herné servery.

Dizajn webovej stránky je moderný, responsívny a na jeho vytvorenie sme potrebovali len zavolať pár tried z CSS frameworku Twitter Bootstrap.

Nie všetko išlo ako po masle a narazili sme na veľké množstvo problémov hlavne pri vytváraní herných serverov, ktoré sme vďaka našim skúsenostiam s programovaním a tvorbou webových stránok vyriešili v rozumnom čase. Vo všetkom zlom je aj niečo dobré, vďaka týmto problémom sme sa obohatili o obrovské množstvo nových skúseností.

V konečnom dôsledku mi táto práca poskytla veľa nového.

## 7. Resumé

قد اخترت هذا المشروع أساسا لأنني رأيت تحديا كبيرا في ذلك. والقصة وراء ذلك، أن هذا المشروع هو عبارة عن موقع خادم للعبة والناس الذين يزورون هذا الموقع يمكنهم تكوين وبدء لعبتهم الافتراضية الخاصة واللعب عليها مع الأصدقاء. أنا صنعت كل من تصميم وجوهر هذا الموقع مع الإطار جانغو (Django) وتويتر بووتستراب (Twitter Bootstrap). أثناء العمل وقعت في العديد من المشاكل مع التشغيل الأمثل ولكن مهاراتي في علم اللوجارتميات ساعدني على حلها. تم إنشاء هذا الموقع كمشروع نهاية العام المدرسي وتخرجي منها، وسوف اخرجه للعلن كمصدر مفتوح بحلول نهاية شهر مايو عام 2015. والنسخة النهائية من هذا الموقع ستكون بالتأكيد مختلفة عن الوثائق، والسبب هو الحصول على افضل تشغيل أمثل وأفضل تحديث وتحسين.



## **8. Bibliografia**

### **Literatúra**

[1]: Greenfeld, Daniel & Roy, Audrey.: Two Scoops of Django: Best Practices For Django 1.6; Two Scoops Press; 2nd edition (February 1, 2014): s.446. ISBN 978-0981467306

### **Internetové stránky**

[2]: <http://www.tangowithdjango.com/book/index.html>

[3]: <https://docs.djangoproject.com/en/1.7/>

[4]: <http://getbootstrap.com/>

[5]: <http://django-simple-captcha.readthedocs.org/en/latest/>

[6]: <https://try.github.io/>

[7]: <http://onody.eu/>

## 9. Obrázková príloha

Obrázok [1.1]:



Obrázok [1.2]:



Obrázok [2]:

## V-Servers.eu - free GameHosing



Obrázok [3]:

## Vyberte si hru!



Obrázok [4]:



Obrázok [5]:

[V-Servers.eu](#) [Vytvoriť server](#) [O nás](#) [Blog](#)

# Blog V-Servers.eu

Miesto, kde sa dozviete informácie a plány k tomuto projektu.

## Hello World!

Vitajte v tomto super systéme ktorý je poháňaný Django-m. Toto je váš (teda môj) prvý článok. Môžete ho upraviť alebo vymazať a potom už len začať písať!

Comments

Community

Sort by Best ▾

Share Favorite

Start the discussion...

ZoltanOnody Mod

• a few seconds ago

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce vel nunc bibendum nisi tincidunt lacinia efficitur at lectus. Suspendisse euismod metus vitae neque varius ultrices vitae et arcu. Nunc efficitur lacus dui, eu placerat nibh scelerisque id. Vestibulum feugiat lectus eget nisi lacinia, id pretium nisi venenatis.

Donec mollis ac nunc sed ornare. Fusce pulvinar lobortis convallis. Pellentesque dapibus commodo molestie.

^ | ▾ • Edit • Reply • Share ▸

Kategórie


[Všeobecné](#)

Značky

[Informácie](#)

DISQUS

© 2014 - All right reserved - just kidding, no rights reserved

 **servers**  
www.v-servers.eu

- 27 -