CampPlanner – dokumentace

Jiří Setnička

CampPlanner je systém pro plánování časového rozvrhu a sledování potřebného materiálu a úkolů při plánování akcí typu tábor či soustředění. Je realizován jako CGI modul v jazyce Perl a jeho rozhraním je tedy webová aplikace.

Centrálním bodem CampPlanneru je samotný rozvrh, ve kterém se dají přidávat, editovat či odebírat jednotlivé položky rozvrhu (hry, přednášky či jiné aktivity, dále referováno jen jako *hry*). U každé hry je možné nastavit několik parametrů (podle kterých se poté zobrazuje v rozvrhu či se dá filtrovat) a také přidat popis (v současné verzi formátový pomocí Markdown syntaxe).

Umístěním do rozvrhu vznikají jednotlivé časové sloty hry. Existují tři různé typy časových slotů: primární slot, kdy se hra přímo odehrává, a sloty na přípravu a úklid hry. Ke každému typu časového slotu lze přiřazovat organizátory a podle tohoto přiřazení pak lze pro každého z organizátorů zobrazit jeho individuální rozvrh.

Posledními věcmi, které lze přidávat ke hrám, jsou úkoly a potřebný materiál. Materiál i jednotlivé úkoly lze označovat za splněné (obstarané) a oboje lze vypsat jako přehled napříč všemi hrami.

Celý systém je vybaven systémem pro sledování změn. Je to odlehčená varianta plnohodnotných verzovacích systémů, nelze se totiž vracet do minulosti a slouží tedy jen ke generování emailů s diffem oproti předchozí uložené verzi. Je realizován tak, že ke každé hře se generuje její výpis do textového formátu, tento se pak ukládá s každým committem do databáze a při uložení nové verze se provede diff mezi poslední verzí uloženou v databázi a současným stavem.

Použití

CampPlanner v současnosti používá několik dalších Perlových modulů specifických pro potřeby Korespondenčního semináře z programování, jedná se hlavně o moduly `UCW::CGI`, `KSP::CGI` a `KSP:DB`, z čehož poslední dva jmenované se dají při troše práce vyměnit za jiné (zmiňované moduly poskytují přesměrování a přístup k databázi).

Pro použití tedy stačí CampPlanneru poskytnout připojení k databázi (tabulky databáze v příloze A) a inicializovat ho následující posloupností příkazů:

```
my $planovac = new KSP::CampPlanner('sksp2014');
$planovac->setDays('2014-09-13', 8);
$planovac->prepare($user);
start_html();
$planovac->show();
```

kde `start_html()` je vypsání HTML layoutu stránky před obsahem vypisovaným CampPlannerem (`CampPlanner::prepare()` totiž obsahuje kód, který ještě může stránku přesměrovat, a proto je nutné aby zahájení výpisu HTML bylo až po něm). Proměnná `$user` pak obsahuje hash představující přihlášeného uživatele a obsahující následující položky: `{id: 1, name: "Jméno uživatele"}`.

Nastavení

V základní ukázce použití již bylo zmíněno volání `CampPlanner->setDays()`, které přijímá dva parametry: počáteční datum a délku akce ve dnech (včetně počátečního dne). Toto nastavení je nutné a bez něj se tato instance CampPlanneru odmítne spustit.

Další nastavení je možné pomocí volání `CampPlanner->setTimeslots()`, kterému lze předat hash jednotlivých částí dne (ve formátu klíč a název každé části dne – je navíc nutné aby jednotlivé klíče byly pojmenované tak, aby po jejich lexikografickém seřazení odpovídaly zamýšlenému pořadí částí dne). Pokud se toto nenastaví, použije se dělení dne na osm částí (ráno až noc, viz zdrojové kódy).

Ostatní nastavení (například emailová adresa na kterou se mají posílat diffy) je pak přímo součástí zdrojového kódu.

Programátorská dokumentace

Základní členění

Celý modul je dle schématu MVC rozdělen do tří souborů: základní `CampPlanner` a poté `CampPlanner::View` starající se o vypisování HTML a `CampPlanner::Versions` starající se o verzování.

Hlavní soubor se stará o přijmutí všech HTTP GET a POST požadavků a podle nich pak rozhoduje o další akci. Akce může být buď odpovídající zpracování dat, modifikace databáze a přesměrování (typická reakce při HTTP POST požadavcích), nebo zavolání odpovídající funkce z `CampPlanner::View`, která se postará o vypsání dat jí předaných.

Z hlediska přístupu k databázi byl zvolen přístup zpracování většiny věcí na straně Perlu. Modul se zeptá databáze na všechna data související s danou akcí a na nich pak všechny další operace provádí sám (namísto zatěžování databáze sérií joinovaných dotazů). Tento postup byl zvolen z několika důvodů.

Zaprvé, s ohledem na to, že jedna akce nebude v databázi obsahovat příliš mnoho záznamů, aby byl výkonový rozdíl zpracování v databázi a na straně modulu rozpoznatelný. Naopak akcí v databázi bude s časem přibývat a při mnoho akcích v databázi by nějaký komplikovaný join přes celou tabulku (kde bude většina dat k jiným akcím) představoval zbytečnou zátěž databáze. Druhý důvod byl, že v Perlu se mnohem snáze popíší některé vztahy, omezení nebo spojení různých dat.

CampPlanner

Jak bylo popsáno výše, jedná se o hlavní logickou část celé aplikace. Jeho fungování by se dalo rozdělit do čtyř základních částí:

- Načtení parametrů, načtení databáze – funkce `prepare()`, `loadDatabase()` a přidružené pomocné funkce. Na zpracování parametrů se používá externí funkce `UCW::CGI::parse_args` a všechny načítané parametry jsou uvedeny na vršku funkce `prepare()`. Zde se rovněž provádí jejich kontrola a je tak zabráněno propašování nežádoucích hodnot dovnitř modulu.
- Zpracování parametrů a případný update databáze a přesměrování – stále ve funkci `prepare()`.
- Kontrola změny ve hrách: s pomocí `CampPlanner::Versions` jsou zkontrolovány modifikované hry a případně jim je nastaven flag v databázi. Toto se provádí proto, aby nebylo nutné provádět diff vůči databázi při každém zobrazení rozvrhu, ale aby stačilo provést diff jen ve chvíli, kdy mohlo dojít k nějaké změně.
- Zobrazení odpovídající části uživatelského rozhraní pomocí `CampPlanner::View` – ve funkci `show()`.

CampPlanner::View

Tato část modulu se stará o vypsání jí předaných dat. Sestává se z několika funkcí starajících se vždy o výpis jednoho pohledu uživatelského rozhraní. Možné pohledy jsou:

- Rozvrh her – Ve formátu tabulky se spojováním navazujících slotů her pro větší přehlednost. Lze filtrovat rozvrh jen pro vybraného organizátora nebo zvýraznit sloty jen vybrané hry.
- Rozvrh volných organizátorů – CampPlanner si udělá seznam všech použitých jmen organizátorů, jejich alokací ke slotům a na základě toho v každém slotu zobrazí organizátory, kteří nejsou přiděleni k žádné hře, úklidu ani přípravě.
- Editace rozvrhu her – Existuje ve dvou verzích: Ve fallback verzi (bez nutnosti použití javascriptu) má formu editace slotů jen pro jednu konkrétní hru pomocí checkboxů. V plné verzi se editace provádí pro všechny hry najednou pomocí jejich přetahování Drag&Drop metodou (viz externí knihovny níže).
- Detail hry – Zobrazuje podrobnosti hry a umožňuje přidávat organizátory do slotů hry či je zase odebírat a to samé pro materiály a úkoly dané hry.
- Editace hry – Editace popisu a podrobností. Pro zápis popisu se používá Markdown syntaxe a při použití javascriptu je zobrazen pomocný editor s možností přímého náhledu.
- Seznam her, materiálů a úkolů – Seznamy s možností filtrování.

Externí JS knihovny a styly

Pro usnadnění práce s uživatelským rozhraním jsou použity následující JS knihovny:

- jQuery – Obecný pomocník pro ostatní knihovny (http://jquery.com/)
- EpicEditor – Vložitelný javascriptový Markdown editor. Umožňuje hlavně online náhled vygenerovaného HTML bez nutnosti neustálé komunikace se serverem. Pracuje v normálním módu (kde se náhled přepíná kliknutím na ikonu) a také ve fullscreen módu, kde se náhled HTML zobrazuje na jedné polovině obrazovky (http://epiceditor.com/).
- REDIPS Drag – Knihovna poskytující přizpůsobitelný Drag&Drop interface pro přetahování položek mezi buňkami tabulky. Využívá se pro snadnou a rychlou editaci her v rozvrhu (http://www.redips.net/javascript/drag-and-drop-table-content/).

Další pomocné soubory pak jsou `camp_planner.js` a `camp_planner.css` obsahující volání výše zmíněných JS knihoven respektive styly pro zobrazení celého CampPlanneru.

Příloha A – Databáze

```
CREATE TABLE camp_planner (
        plan_id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
        camp VARCHAR(10),               -- Identifikace akce, napriklad "jksp2012"
        type ENUM('game','lecture','other'),
        name VARCHAR(255),
        description TEXT,
        place ENUM('inside','outside'),
        chosen BOOL,                    -- Je hra vybrana? (pro zobrazeni u rozvrhu)
        modified BOOL                   -- Jsou zde nejake necommitnute zmeny?
);
CREATE TABLE camp_planner_commits (
        commit_id INT NOT NULL,
        plan_id INT NOT NULL,
        uid INT NOT NULL,
        commit TEXT,
        commit_message TEXT,
        time DATETIME,
        FOREIGN KEY (plan_id) REFERENCES camp_planner(plan_id) ON DELETE CASCADE,
        FOREIGN KEY (uid) REFERENCES logins(uid), -- Odkaz na tabulku uživatelů
        UNIQUE (commit_id,plan_id)
);
CREATE TABLE camp_planner_tags (
        plan_id INT NOT NULL,
        tag VARCHAR(20),
        FOREIGN KEY (plan_id) REFERENCES camp_planner(plan_id) ON DELETE CASCADE,
        UNIQUE (plan_id,tag)
);
CREATE TABLE camp_planner_timeslots (
        plan_id INT NOT NULL,
        type ENUM('primary','preparation','cleanup'),
        timeslot VARCHAR(30),           -- Identifikace slotu
        FOREIGN KEY (plan_id) REFERENCES camp_planner(plan_id) ON DELETE CASCADE,
        UNIQUE (plan_id,type,timeslot)
);
CREATE TABLE camp_planner_orgs (
        plan_id INT NOT NULL,
        type ENUM('garant','assistant','primary','preparation','cleanup'),
        orgname VARCHAR(30),
        FOREIGN KEY (plan_id) REFERENCES camp_planner(plan_id) ON DELETE CASCADE,
        UNIQUE (plan_id,type,orgname)
);
CREATE TABLE camp_planner_materials (
        material_id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
        plan_id INT NOT NULL,
        name VARCHAR(255),
        note TEXT,
        completed BOOL,
        FOREIGN KEY (plan_id) REFERENCES camp_planner(plan_id) ON DELETE CASCADE
);
CREATE TABLE camp_planner_todos (
        todo_id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
        plan_id INT NOT NULL,
        text TEXT,
        completed BOOL,
        FOREIGN KEY (plan_id) REFERENCES camp_planner(plan_id) ON DELETE CASCADE
);
```