PythonQuest aims to make learning the basics of programming with Python easy and fun.

PythonQuest started as a final project for WWW-sovellukset -course at LUT.

Technical features were priority before the deadline which means the game isn't playable just yet.

---Installation---

Project is developed with Apache, MySQL and Memcached
Languages: PHP, HTML, CSS, Javascript (with jQuery and Phaser)

1. Get the source (i.e. git clone https://github.com/toivomattila/PythonQuest.git)
2. Set up Apache to point to PythonQuest
3. Set up database:

    -Edit models/utils.php if database is no running on 127.0.0.1

    -Either edit models/utils.php or create a user with username and password from mo
dels/utils.php

    -Import PythonQuest.sql
4. Set up memcached:

    -Edit models/utils.php if memcached is not running on 127.0.0.1

---Points---

Total 47

Implemented features

5 Authentication (User can log in and sign up, code is in models/signup.php, models/login
.php)

5 Database used to store data (Authentication uses database)

5 MVC-design pattern (Folder structure, view and controller are somewhat coupled, model t
rys to be decoupled from both)

5 Responsive layout (The website responds to window size changes except for canvas which
is fixed size)

5 Documentation (this file & comments in the code)

3 Canvas-element (the game is drawn to canvas)

3 JSON for moving/storing data (game/models/model.php returns JSON data)

3 Ajaj/Ajax (game/controllers/fightController.js uses Ajax to get data from server with P
OST)

3 jQuery (For animation on signup, checks if password is okay)

3 Front-controller (Everything goes through index.php)

3 Memcached (Used to store i.e. player data in game/models/model.php)

2 Browser compatibility check for canvas (views/game.html has a script that checks for ca
nvas & pushes text to html if not supported)

2 Animation with jQuery (during signup, code is in views/signup.js)