Team Project

JavaScript UI & DOM 2014 - online

Team Mojo Jojo

Members (Name / Telerik username / GitHub username):

1) **Галя Богданова - Galya - Enlightment**
2) **Теодор Върбанов - tvarbanov - tvarbanov**
3) **Николай Куманов - NP.Kumanov - NK-Hertz**

Respository: https://github.com/MojoJojoTelerik/RPGCanvas/

Demo: http://www.galyabogdanova.com/projects/RGPCanvas/index.html

Game

NOTE: On Chrome browser the game is working on live server, but not on localhost due to Chrome bug where the browser considers file:// path as url to another domain (cross-domain security issue). Please take a look at our live demo or use another browser.

Използвани технологии:

- **KineticJS**
- **Raphaël JS**
- **JQuery**

Използвани анимации:

KineticJS:

- Анимиране на изстрелите (ShotFactory.js) -> shot.animate = new Kinetic.Animation
- Анимиране на придвижването на играча (Game.js) -> var playerMoving = new Kinetic.Animation
- Анимиране при смяната на sprite images - вътрешна функция за KineticJS, използвана върху Sprite на играча (Stage.js) -> this.playerImage = new Kinetic.Sprite;

Raphaël JS:

- Анимиране на полето, което показва текущото състояние на живота на играча (gameStats.js) -> lifeBar.animate

Цел на проекта: Да се създаде RPG игра за браузър.

Геймплей: Със стартиране на играта животът на героя започва постепенно да се намалява с по една точка на определен интервал от време. Отчитат се редица колизии с различни обекти, които повлияват движението на играча. На определен интервал от време на случайни позиции се създават enemies. Когато играчът удари enemy, си връща 50 точки към живота. Когато точките на живота достигнат 0, играта свършва.

Контроли:

- Space - изстрел
- Arrows - движение в съответната посока

Логика на приложението:

След зареждане на обектите от DOM дървото, се извиква ивент от index.html, в който се дефинира JSON обект с имената на всички изображения, необходими в играта,

и пътя до техните файлове. JSON обектът се подава като аргумент на функцията loadImages(), която първо **зарежда асинхронно изображенията в браузъра** чрез KineticJS event-а image.onload, **след което** подава асоциативен масив със заредените изображения като аргумент при **създаването на обект от тип Game**, т.е. при стартирането на играта.

В Game последователно се създават няколко обекта: stage, shotFactory, enemyFactory, player, obstacles, gameSvgStatistics и се извикват няколко функции, сред които eventsStart(), съдържаща **необходимите събития** за проследяването на keyboard input и за придвижването на играча. В Game.js се съдържа още JS Interval mainLoop, който ъпдейтва на определен интервал от време **колизиите** на изстрелите с другите обекти в играта, както и дали са изпълнени условията за край на играта.

Инстанцирането на класа Stage осъществява създаването на KineticJS stage и layers (canvas), както и дефинирането на използваните в играта KineticJS обекти - Images & Sprites, които след това се добавят към съответните layers. Върху някои от обектите се извиква още функцията за кеширане на изображенията, за да може да се използва Kinetic **филтъра за промяна на осветеността** или за да могат **прозрачните пиксели** на изображението да бъдат **игнорирани при засичане на колизия** (в случая със скалата). Създава се JSON обект с всички необходими обекти от stage-а, които ще се използват в логиката по-нататък и той се подава при извикването на главната функция.

Обектите от тип Player и Shot присвояват свойствата на съответстващите им KineticJS Image / Sprite обекти и **разширяват функционалността им**.

Енкапсулация: Само необходимите свойства и методи са публично достъпни! Разчита се основно на подаване на обекти и променливи като аргументи на извикваните функции.