This document describes the **team work assignment** for Telerik Academy students studying **JavaScript Applications**.

# Project Description

Design and implement **an object-oriented web application** by choice. It could be a simple game (e.g. Snake, Chess, Backgammon, Minesweeper, Tetris, Xonix or other), component library (windows, buttons, text boxes, menus, etc.), business application (e.g. car store, auction, movie rental, bug tracker, knowledge management system, e-library, text editor), or any other. You are absolutely **free to choose the topic of your work**.

## General Requirements

Please define and implement the following assets in your project:

* **Use jQuery** (DOM manipulation, AJAX, animations, etc…)
* **Implement OOP design** (implement the application logic using objects, modules and data hiding)
  + At least **3 modules** (namespaces)
  + At least **7 types of objects** ("classes")
* **Unit testing for all of the components**
  + Using Jasmine, QUnit or JsUnit
* **Implement a UI for your application** (use KendoUI, jQueryUI or implement your own UI logic)
* **Use some kind of data storage** (localStorage, sessionStorage, SQLite, localDB, REST services, etc…)
* **At least one third-party API to share something from your application**
  + Share status to Facebook, Twitter or Google+
  + upload images to Facebook or Flickr
* **The application must work in Google Chrome 26, Mozilla Firefox 21, Internet Explorer 9, Internet Explorer 10, Opera 12**

## Additional Requirements

* Follow the **best practices for OO design**: use data encapsulation, use exception handling properly, use inheritance, abstraction and polymorphism properly and follow the principles of strong cohesion and loose coupling.
* Obligatory use **Subversion (SVN)** to keep your source code and for team collaboration (you might use [http://code.google.com](http://code.google.com/), [http://projectlocker.com](http://projectlocker.com/) or [https://www.assembla.com](https://www.assembla.com/)). TFS or Git are **not** allowed. Use SVN.

## Optional Requirements

If you have a chance, time and a suitable situation, you might add some of the following to your project:

* **Backward compatibility** (make the application usable on browsers like IE8, IE7 and IE6)
* **Usage of a structural JavaScript framework**
  + Backbone.js, Ember.js, AngularJS or Knockout

## Non-Required Work

* **Completely finished project** is not obligatory required. It will not be a big problem if your project is not completely finished or is not working greatly. This team work project is for educational purpose. Its main purpose it to experience **object-oriented modeling** and **OOP** in a real-world project and to get some experience in **team working** and team collaboration with TFS.
* **Implementation of server-side logic with ASP.NET WebForms, ASP.NET MVC or PHP**

## Deliverables

Put the following in a **ZIP archive** and submit it (each team member submits the same file):

* The complete **source code**.
* Brief **documentation** of your project (2-3 pages). It should provide the following information (in brief):
  + Team name and list of team members
  + Project purpose – what problem do you solve?
  + Class diagram of your types
  + The URL of your SVN repository
  + Any other information (optionally)
* Optionally provide a **PowerPoint presentation** designed for the project defense.

## Public Project Defense

Each team will have to deliver a **public defense** of its work to the other students and trainers. You will have **only 5 minutes** for the following:

* **Demonstrate** the application (very shortly).
* Show the **class diagram** (just a glance).
* Show the **source code** in the **SVN-**web-based source code browser.
* Show the **commits logs** to confirm that team member have contributed.
* Optionally you might prepare a PowerPoint presentation (3-4 slides).

Please be **strict in timing**! Be **well prepared** for presenting maximum of your work for minimum time. Bring your own laptop. Test it preliminary with the multimedia projector. Open the project assets beforehand to save time. You have **5 minutes**, no more.

## Give Feedback about Your Teammates

You will be invited to **provide feedback** about all your teammates, their attitude to this project, their technical skills, their team working skills, their contribution to the project, etc. The feedback is important part of the project evaluation so **take it seriously** and be honest.

NS = namespace

# Game namespaces and objects

#### GameObjectsNS (drawable or not?)

* GameObject
  + Type – circle, rectangle, ….
  + Color – predefined string object
  + Coordinates
  + Update Object function – move, get eaten etc.
  + Collision Group
* MovingObject
  + headObject
    - speed / or direction
    - can collide with staticObjects, movingObjects
    - canHaveTail(has oldCoordinates field that will be given to tail)
  + tailingObject
    - following previous tail or head( takes Coordinates from previousElement.oldCoordinates)
* snakeObject – maybe in controllers???
  + has head and tailing objects (inside an array like object)
  + update updates all its objects and moves them…. Or
* StaticObject
  + stoneObject
  + wallObject
  + foodObject
* gameObjectsFactory - instantiates object – moving/static with given name or

#### GameControllersNS

* gameController
  + speed of game (level diff….?) at construction time
  + listOfObjects
  + collisionDetector
  + drawer (canvas) - instance
  + userInteractor – handles user input
  + main game loop
    - collisionDetection
    - objectsUpdate
    - check if game over
    - spawn food?
    - objects draw
* menuController
  + creates buttons
  + gives eventlisteners
  + visualizes menu
  + maybe has background????
* (Snake Controller??? Here ??? where???)

#### CanvasDrawerNS

* canvasDrawer – takes an object and draws its representation on canvas
* menuDrawer – takes menu object

MenuNS(?? or inside GameObjects – menuObject)

## GameFlow

1. Draw menu(initialize controller (or maybe menuController) – initialize drawer, initialize menu object)
2. Start menu….. process input ….. – highScore, startGame, difficultyLevel, about Team “Maragedik”, exit (byeBye)
3. Start
   1. startGame
      1. instance of objectsFactory, initialize gameController if none
      2. createGameField – walls, food, head ( with 2 tailingobjects)
      3. startMainLoop -
   2. highScore
      1. clearCanvas
      2. getScores
      3. draw scores ( in a DOM element or using canvas drawer?)
      4. goto 2. (goto is evil)
   3. about
      1. take information for team from facebook!!!! API
      2. Take information about user’s facebook – using facebook API
   4. Exit
      1. Bye bye – close window

### Main Loop

Main game loop :

1. Collision detection
   1. For each element in the elements collection check if collided and take action
2. Objects Update
   1. For each object update
3. check if game over
   1. if over take action
4. spawn food?
   1. Random generate food or when eaten…
5. objects draw
   1. draw all objects ( for each object in game objects collection execute