Web Development

European Competition 2021

TEST PROJECT

Final version

SKILL 17 web tehnologies

Test Project: Game

Time: Six hours

The *Winter Sports Popularization Association* (WSPA) wants you to develop an online game. In the game, our hero The snowMan 2.0 will spend some quality time outside and show how fun it can be to have friendly snowball throwing battles.
The goal is to throw an object (hereafter the projectile) as far as possible.

Design and layout

Several game assets are provided and should be enough for you to develop our game. However, our designer is stubborn and keeps making it all green and blue, but our brand colours are defined in the style guide matching with the WSPA logo. Please use correct colours and include the WSPA logo to start and end screens of the game.

The game should run on both desktop and mobile. The dimensions of the game area are a minimum of 1000 x 640 pixels on desktops. In mobile, the game should zoom or resize to fit the device width and keep the same aspect ratio.

On the desktop, the game is controlled via keyboard. On mobile, the game is controlled by touching the screen area.

Your task is to create a game screen based on the assets given to you. You can add your own elements to enhance the game layout.

Assets format

The game assets format should be optimised for loading performance. Please use WebP or SVG for all the images and provide a fallback format only to browsers that do not support WebP.

There should also be at least 3 sound effects in the game.

1. Game screen after loading finished.
2. When hitting the projectile.
3. When the game is over.

You are free to add more sound effects. Please also use the file format and fall backs to support major browsers.

Start screen

On the first screen, the player needs to enter his/her name and select the 2-letter country code from a list (https://restcountries.eu/). The name and country inputs are mandatory fields.

To make the game more personal, the player can upload the projectile image. The player can either choose to upload via clicking the upload button or drag and drop the image into a dropping area. The dropping area should be visible and clear to the player. The uploaded image can be .png, .webp, or .jpg format. Make sure that the uploaded image is square. Other file formats are not allowed to upload. A default projectile will be provided if the user does not upload an image.

Every game played should be stored in a database. Create an API call: /api/start-game, and store all players data from the start screen. Keep in mind that during and at the end of the game, you will have key moments for updating the game session that is being played.

Loading screen

Not all our community members have a fast internet connection. Please make sure all game assets are pre-loaded with JavaScript after pressing Start on the start screen but before the game starts.

For a better user experience, visualise the loading process, so the player understands what is happening. Please provide loading progress on how many assets loaded and how many are waiting to be loaded. Please provide any kind of animation to indicate that the loading is not freezed.

The snowball throwing scene

Our local hero, The snowMan 2.0 to throw the projectiles. He is on the left-hand side of the scene and he is throwing projectiles to the right side as far as he can.

For each game session, there are 3 projectiles for the player to throw. After each throw, the projectile rests on the destination as an indicator. Think of it as a ghost car in a racing game that shows the previous record but does not affect the game play.

The user can control two parameters (one at a time) to give the projectile a good throw.
1. Angle
2. Power

The angle has a vertical bar going up and down and the power has a horizontal bar going left to right every e.g. *0.25* seconds. Find the right amount of time to make the game more fun to play. Both bars should show an indication of the scale. For example, it may be a gradient heat map or spectrum scale to indicate from worst value to best.

Game Control

The player either hits the spacebar button or taps on the screen on a mobile device at the right moment to stop the two parameters, one at a time.
This is done at the moment when the player thinks that angle and power are right, so that the projectile would achieve the most distance.

If it is the optimal combination, the game freezes at the moment of the hitting touch for 1 second, and shows a "perfect" asset to enhance the visual effects. This only happens when hitting the perfect combination. During game debugging, there should be a way to trigger this "perfect" hitting effect for any combinations of angle and power. To assess this “feature” a “debug-mode” button is included, which triggers this functionality.

We have created and provided an example implementation that you could use (and may even improve it) to determine the journey of the projectile (see Game Mechanics)

After selecting the angle and the power, the projectile starts flying from left to right on a 2D scene, but the projectile is focused in the middle of the scene.

This means our snowman and the background scenery will move towards the left side and out of the game area on the left side. You need to implement indicators for showing the distance of the throw.

When the projectile flys through the air, a particle visual effect is shown at the tail of the projectile.

It is important to save the exact moment of the throw to the user’s game session into the database. Saving the game starting timestamp, angle, and power of the attempt.
Create an API call: /api/save-throw.

Game Over conditions

There is only one game over condition:

- After all 3 attempts have been played

Attempt failed conditions:

- When a projectile goes out of the screen on the top side;
- When the angle is less than 5 degrees and power less than 30%;

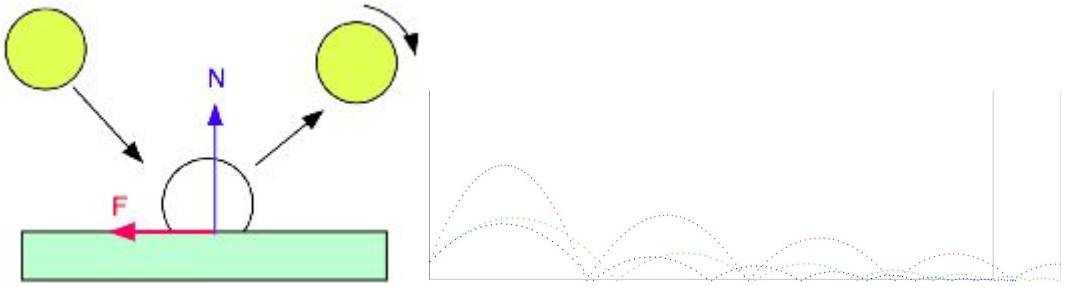
When the attempt is over, the projectile's speed is set to 0 and this attempt is saved to the database for the current game. Create an API call: /api/game-over

Game mechanics

While the projectile is flying, its course is calculated by our formula, and the trajectory is saved into the database every 0.5 seconds also at the same time the speed of the projectile is reduced by 10%. This way we create a log of the throw. Create an API call: /api/in-flight and store in the database the speed and x and y coordinates of the projectile.

Bounce(s)

At some point, the projectile will come back to the ground, and depending on whether it still has enough speed (at least 20% of the initial speed), it might bounce off the ground and add a bit more distance.
When the projectile's speed is still above 60% of the initial speed, let the projectile bounce and rotate 90 degrees forwards.



The bounce-back angle should be calculated based on the incoming angle minus 10%.

If there is enough inertia (speed) on the next projectile landing, the bounce is repeated.

Every bounce should be stored in the database to the current game session. Create an API call: /api/bounce

If the projectile's speed is below 20% of the initial speed, initiate the final glide of 0.5 seconds, and reduce the speed to 0.

Here is a sample implementation of the formula which has been mentioned. Feel free to use and adapt it accordingly: https://codesandbox.io/s/magical-sunset-jn662

End screen

We should see the end screen when the game over conditions are met.

Here is how to calculate the score for this entire game session. After the game ends, there are a total of 3 projectiles thrown. The longer distance the better. The game takes the average of the top 2 distances as the final result.

Show a leaderboard of the five top players. Display the name, country code, the image of the projectile used, and their results on each row.
Clearly highlight the position of the current player's recent game session and the top three positions within the list. If the current player's position is out of the "top 5 players" it is displayed as an additional row.

There is also a button that brings you all the way back to the start screen, where the player can either start a new game or play again with the same credentials (name, country code and projectile image).

Result Sharing

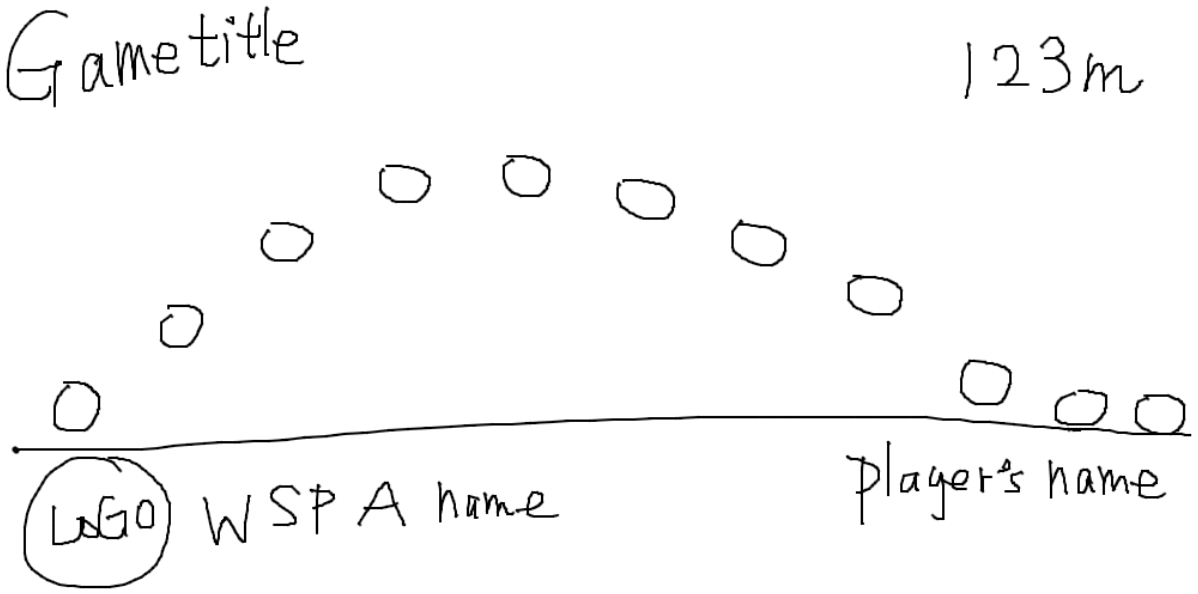
The goal of building this game is to promote our client, The *Winter Sports Popularization Association* (WSPA). We want to let the player share the final result as an image, so that they may further share it to their friends on social networks.

In the end screen, the following image is automatically generated and displayed to the player. A horizontal image that displays the best result of the three attempts with the path of the projectile.

The client wants the following elements to be on this shareable image:

1. The text *Winter Sports Popularization Association*.
2. The logo for the WSPA.
3. The player's name.
4. The path of the projectile, using the correct image.
5. The distance.

Here is an imaginary result shareable image, the layout is free to be adjusted.



Submitting the work

All work should be stored on the provided server. If the CLI JS version is used, there should be a “js-src” folder with an uncompiled JavaScript version.

The game needs to be accessible and playable through the following URL:

https://<yourhost>.comp.skill17.com

Tools

What competitors can use

- Competitors are allowed to use the Internet.
- Competitors are allowed to use their own computer.
- Docker images have been provided for the competitors to work on.

Communication Tools

- The main visual communication channel will be provided in TP brief.
- In case of any technical issues with the server please write in the WhatsApp channel.