Drawing game

Darjan Babic, Martin Tobias Klug, Christoph Ladler, Lukas Wagner

WebApp

- Drawing game
- Multiplayer
- One draws, others guess the word
- Ranking system (points according to how much time is left)
- Words to draw are randomly chosen from a wordlist that is selected at the game creation

Architecture

- client - server architecture and Model-View-Controller
  - View: Canvas, High score list
  - Controller: Chat answers, Canvas drawing, server communication
  - Model: High score list storage, Word list
- client:
  - display canvas
  - chat for guesses (correct guesses are not shown)
- server:
  - session/game management
  - message transfer
  - canvas synchronisation

HTML 5 Features

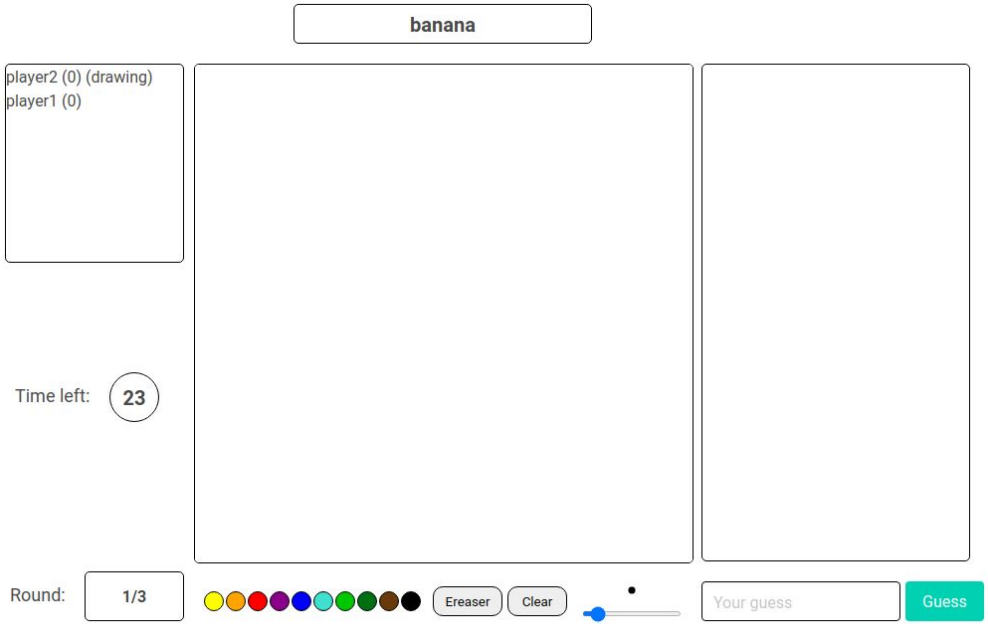
- 2D canvas
- Web sockets for client-server communication
- Web storage for word lists
- Drag and Drop
- Audio elements during game

Audio elements

- ticking of clock (gets faster when time is almost over)
- when a new round starts
- message sent
- player guesses the word
- at the end of the game

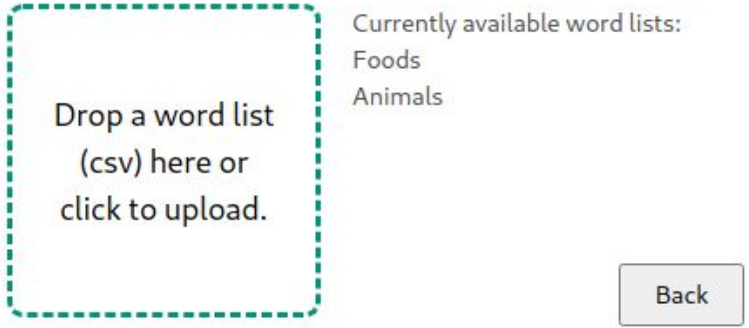
2D Canvas

- drawing element
  - adjustable pen size and color
  - clear button and eraser for drawing
- uses mouse events
  - “mouseenter”
  - “mousedown”
  - “mousemove”
- use Canvas specific commands
  - “getContext”
  - “stroke”



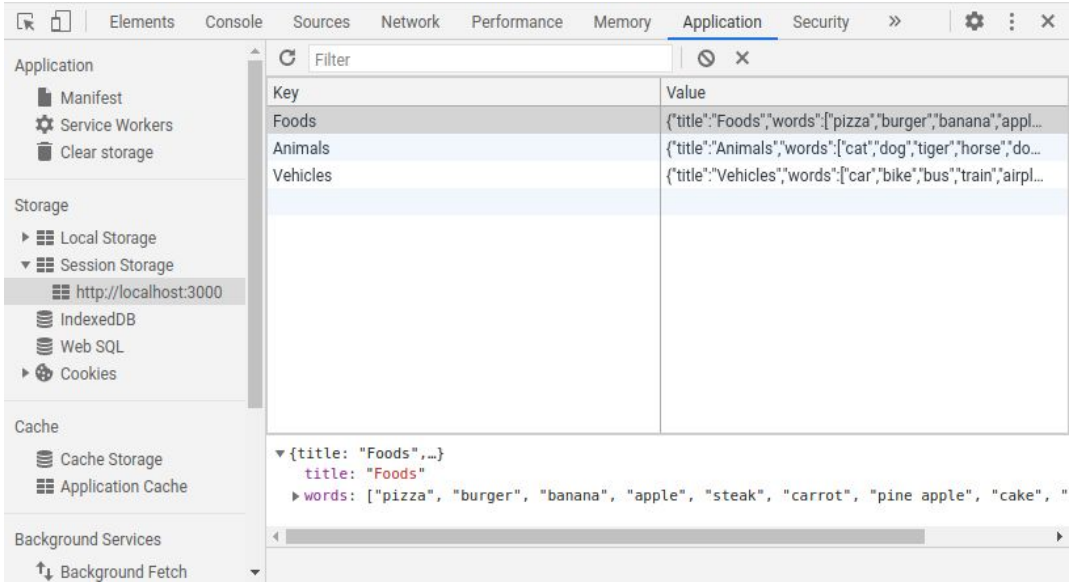
Drag and Drop

- import custom word lists
  - csv format
  - title of file is the title of the word list
- default behaviour needs to be prevented
- uses different drag events
  - “dragover”
  - “drop”
  - “dragleave”



Webstorage

- Word lists
- localStorage
  - persists if browser is closed
- sessionStorage
  - maintains storage for each given origin
- storage limit larger than cookie (5MB)
- accessible via Window object



Server

- Implemented in JavaScript
  - Node.js (runtime environment for server-side JavaScript)
  - express (web framework)
  - express-ws (extension for express used for web sockets)

Server API

- REST API for game/wordlist management
- WebSocket API for real time communication during a game
- JSON as a data format for both

REST API

- `GET /api/wordlists`
  - Fetch server provided pre-defined word lists
- `POST /api/game`
  - Create a new game
- `GET /api/game/<gameId>`
  - Fetch information about existing game
- `POST /api/game/<gameId>/join`
  - Join a game

WebSocket Messages

- `HELLO` → Handshake with given user identifier
- `READY_STATE` → Signal whether user is ready or not
- `CANVAS_CONTENT` → Distribute canvas content
- `CHAT_MESSAGE` → Word guesses in chat area
- `USERLIST_UPDATE` → Information about users (joined, points, ready, etc.)
- `ROUND_UPDATE` → Sent for every new round
- `GAME_UPDATE` → Update word/word hint, clock

Demo