

# Élaboration d'un jeu multijoueur en HTML5

Florent Marchand de Kerchove

Université du Havre

27 juin 2011

# Outline

## 1 Technologies fondatrices

JavaScript

Canvas HTML

WebSocket

Node.js

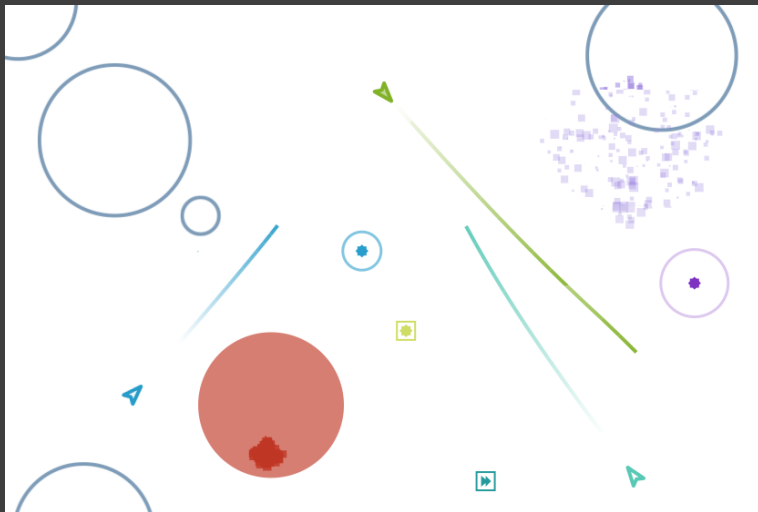
## 2 Derrière Spacewar

Principe de jeu

Côté client

Côté serveur

# Intro



[Glorious screenshot !] [Rationale]

## 1 Technologies fondatrices

JavaScript

Canvas HTML

WebSocket

Node.js

## 2 Derrière Spacewar

Principe de jeu

Côté client

Côté serveur

# Historique de JavaScript

- Langage de programmation des navigateurs web
- Créé en 1995 par Brendan Eich de Netscape
- Standardisé sous le nom d'ECMAScript in 19??
- Possibilités limitées au début
- Regain d'intérêt avec Ajax
- Nombreux frameworks puissants (jQuery, CommonJS, Dojo)
- Présent sur la majorité des sites d'aujourd'hui

# Caractéristiques de JavaScript

- Descend de Scheme et Self
- Langage fonctionnel, dynamique
- Faiblement typé, *closures*
- Héritage par prototype
- Syntaxe héritée du C

# Attraits de JavaScript

- Langage de haut niveau
- Multi-plateforme (JVM sans installation)
- Fonctionnalités asynchrones
- Programmation évènementielle

# Inconvénients de JavaScript

- Tendance verbeuse
- Globales implicites
- Opérateurs d'égalité coercitifs



# CoffeeScript

- Langage intermédiaire qui compile vers JavaScript
- Syntaxe empruntée au Ruby
- Épuré et concis
- Array comprehension, lambda function, bindings, anonymous wrappers ...

# Surface de dessin

- Permet de dessiner et d'animer librement sur une page web
- Contextes 2d et 3d
- Alternative aux SVG plus performante
- Surface *bitmapped* plutôt que vectoriel
- Pas d'insertion dans le DOM
- Accélération matérielle possible



# Protocole WebSocket

- Réponse du standard aux techniques Comet
- Rend obsolète HTTP *long-polling* et HTTP *streaming*
- Véritable *full-duplex* entre client et serveur HTTP
- Mise à jour de la connexion TCP créée pour la requête HTTP
- Protocole en évolution

# Node.js

- Serveur performant écrit en JavaScript
- Entrées/sorties asynchrones
- Programmation événementielle
- Nombreux modules dont Socket.IO

# Exemples de Node.js

- Serveur écho, chat, proxy ...

## ① Technologies fondatrices

JavaScript

Canvas HTML

WebSocket

Node.js

## ② Derrière Spacewar

Principe de jeu

Côté client

Côté serveur

# Principe de jeu

- Jeu d'action frénétique dans l'espace
  - À chaque joueur un vaisseau
  - But : tirer sur les autres et survivre
- Démonstration



# Éléments du jeu

- Contrôles simples :
  - Tourner à gauche, à droite
  - Avancer
  - Tirer
  - Utiliser un bonus
- Carte torique
- Obstacles : planètes et satellites
- Trajectoire des balles affectées par la gravité des planètes
- Les bonus apportent de la variété (mines, turbo, bouclier, ...)

# Rôle du client

- Relayer les entrées claviers au serveur
- Recevoir les messages du serveur
- Afficher le jeu en temps réel

Semblable à un terminal : toute la logique est côté serveur.

# Boucle de dessin

- Un jeu d'action requiert un rendu fluide (40 à 60 FPS)
- Requiert de dessiner très rapidement une image
- Canvas HTML

# Dessiner le tore

Donner l'illusion d'une carte torique

- Remplir le canvas de copies de la carte
- Appliquer la logique au tore
- Considérer les entités les plus proches sur le tore

# Performance

- Ne pas dessiner les objets hors champ
- Sauvegarder les dessins coûteux dans des *sprites*
- Optimisations de bas niveau hors de notre contrôle
- Accélération matérielle du canvas

# Rôle du serveur

- Gérer la logique du jeu :
  - Initialiser la carte de jeu
  - Mouvoir les objets (vaisseaux, planètes, balles, ...)
  - Détecter les collisions entre objets
  - Résoudre ces collisions
- Synchroniser l'information auprès des clients

# Communications clients-serveur

Connexion d'un client :

- Attribution d'un identifiant
- Création d'un objet *Player* associé
- Envoi de tous les objets de jeu

Durant la partie :

- Les clients envoient leurs entrées clavier
- Le serveur broadcast les changements

Déconnexion d'un client :

- Notification aux autres clients
- Libération des ressources associées

# Initialiser la carte de jeu

- Chargement du fichier de préférences :
  - Dimensions de la carte
  - Nombre de planètes à placer
  - Taille des planètes, des satellites
  - Vitesse et distance des satellites
- Placement aléatoire sans chevauchements



# Boucle principale

Étapes effectuées toutes les 20ms :

- Agir en fonction des évènements clavier
- Déplacer tous les objets
- Détecter et résoudre les collisions
- Récolter les changements d'état de chaque objet
- Broadcaster les changements de tous les objets

# Gérer les collisions

- Traitement symétrique centralisé
- Approche naïve quadratique
- Vérification des collisions entre voisins
- Découpage de la carte en grille

# Améliorations envisagées

- Instanciation des parties
  - Rejoindre une partie aléatoire ou entre amis
  - Création de parties personnalisées
- Communication entre joueurs
- Optimisations serveur
  - Diminuer le coût des collisions
  - Permettre un plus grand nombre de joueur simultanés

## Améliorations envisagées (2)

- Optimisations client
  - Dessiner plus rapidement
  - Améliorer la compatibilité avec tous les navigateurs
- Éléments de jeu supplémentaires
  - Bonus (bouclier, missile)
  - Contenu solo
  - Mesure de progrès (score, statistiques)

Merci

Questions / Réponses