

CanvHunt

Nom de code

Glossaire

Fonctions = Gras

Fonctions du navigateur = souligné gras

Variable = Italique

constantes = *Italique Gras*

État = soulignée

(?) = Cette éléments reste à déterminer s'il sera implémenté et utilisé ou non

Rouge Pas encore commencé

Jaune Commencé, mais non terminé

Vert Terminé

Variables Globale

- **cibles** Contient toutes les cibles sous forme de tableaux
- **score** Contient le score de l'utilisateur
- **qteCibleManquee** Contient le nombre de cible Manquée par l'utilisateur
- **qteCibleInterdit** Contient le nombre de cible
- **t** Contient le timer utilisé pour la fonctions **deplacerCible**
- **isPlay** Contient l'état du jeu (s'il est en pause ou non)
- **isPartieEnCours** Contient l'état de jeu (si une partie est en cours ou non)

Objets (utilisé)

```
Cible = {  
  'x' : 0,  
  'y' : 0,  
  'dimension' : {  
    'hauteur' : 50,  
    'largeur' : 50  
  },  
  'direction' : GAUCHE_DROITE,  
  'vivant' : vrai ou faux,  
  'enJeu' : vrai ou faux  
}
```

- x = Doit être à 0 ou prendre comme valeur la largeur du canevas
- y = Doit être un chiffre entier compris entre la hauteur du canevas – 200 (On se garde une marge pour pas que la cible soit trop basse) Voir la fonction **Math.Random()**

- Dimension
 - hauteur = Contient la hauteur que la cible aura
 - largeur = Contient la largeur que la cible aura
- direction = Contient GAUCHE_DROITE ou DROITE_GAUCHE (Utilisé **Math.random** pour déterminer si la direction de la cible se fera de la gauche vers la droite ou de la droite vers la gauche)
- vivant = Contient l'état de la cible, si la cible est vivant ou non
- enJeu Contient l'état de la cible, si la cible fait toujours partie de l'aire de jeu

Constante Globale

- **MAX_CIBLE_MANQUE** Contient le nombre maximal de cible manqué autorisé
- **MAX_SCORE (?)** Contient le nombre représentant le score maximal autorisé
- **VITESSE** Contient le valeur temporelle pour le setInterval de la fonction **deplacerCible**

Fonctions

- **NouvellePartie** [David]
- **FinPartie** [David]
- **genererCible** [Divine]
- **deplacerCible** [Divine]
- **verifierCible**
- **dessinerCible**
- **gererClic** [Boubakar]
- **togglePlay** [Boubakar]
- **sauvegarderScore**

Informations complémentaires sur les fonctions

- **NouvellePartie** [David]
 - Remets les variables en sont états par défaut
 - appel **genererCible**
- **genererCible** [Divine]

Cette fonction permet de générer les cibles au début de la partie.

 - Les *cibles* auront une valeur Y de départ différente
 - Une *direction* gauche vers droite (**GAUCHE_DROITE**) ou droite vers gauche (**DROITE_GAUCHE**)
 - Un tableau contenant toutes les cibles

- Si un timer (*t*) existe déjà, il ne faut pas en créer un nouveau
- Appel **deplacerCible** à l'aide de la fonction `setInterval` à la fin
- **deplacerCible** [Divine]
 - Si la cible est vivante et en jeu alors on peut la déplacer, sinon on n'y touche pas
 - Si *direction* == **GAUCHE_DROITE** alors on incrémente la valeur *x* de la cible (*x*++)
 - Si *direction* == **DROITE_GAUCHE** alors on décrémente la valeur *x* de la cible (*x*--)
 - appel **verifierCible**
 - appel **dessinerCible**
- **verifierCible**
 - Vérifié si des cibles ont quitté l'aire de jeux, si c'est le cas incrémente *qteCibleManque* et modifier la propriété *enJeu* de la cible
 - Si le nombre *qteCibleManque* est \geq **MAX_CIBLE_MANQUE** alors appel **finPartie** avec le paramètre *reussi* à faux
- **gererClic** [Boubakar]
 - Vérifie si le clic se trouve sur une cible, si c'est le cas alors incrémente *score*
 - Si le *score* est \geq **MAX_SCORE** alors appel **finPartie**
- **finPartie** (*reussi*) [David]
 - Si *reussi* == true alors dessiner le tableau de score avec un message de félicitation
 - Si *reussi* == false alors dessiner le tableau de score avec un message 'Meilleure chance la prochaine fois' ou autre message
- **dessinerCible**
 - Nettoyer le canvas (à l'aide de **clearRect**) et ensuite affiche les cibles
 - si la cible est en jeu alors on la dessine sur le canevas, sinon on ne l'affiche pas
- **togglePlay**
 - *isPlay* == true
Arrête la minuterie *t* avec **clearInterval**
Retirer les gestionnaire d'événement (`removeEventListener`)
 - *isPlay* == false
Démarré la minuterie *t* avec **setInterval** (**deplacerCible**)
Ajouter les gestionnaire d'évènement (`addEventListener`)
 - *isPlay* = le contraire de *isPlay*

Comment créer et interagir avec des objets

```
var cible = {  
  'x' : 50,  
  'y' : 20  
};  
  
var cibles = new Array();  
cibles.push(cible);  
  
//OU  
var cibles = new Array();  
var cibles.push({  
  'x' : 50,  
  'y' : 20  
});
```

```
for (var i = 0; i < cibles.length; i++) {  
  //Affiche les valeurs x et y de la cible en cours  
  //Dans la boucle For  
  console.log(cibles[i].x, cibles[i].y);  
}
```