



**Print it!**  
Impressions d'entreprises

# Carrousel en Javascript

## Récapitulatif des points essentiels



- Créer un slider à l'image du dessus au clic le texte présent ainsi que l'image , et les bullets points devront changer dynamiquement pour suivre les différents textes et images fournis.

## **D**éclarer les constantes afin de récupérer les éléments du DOM

- Commencer par ajouter le script à l' **index.html**
- faire un **console.log('ok')** pour vérifier que tout fonctionne
- Je commence par déclarer les constantes nécessaires à la récupération des id et des classes pour les événements.
- Je déclare donc les flèches avant et arrière, le p , l'image dans laquelle je vais injecter dynamiquement les images au clic, les puces.

```
const btnBefore =  
document.querySelector('.arrow_left');  
const btnNext= document.querySelector('.arrow_right');  
const p = document.querySelector('p');  
const slider_img =  
document.querySelector('#slider_img');  
const puces = document.querySelectorAll('li');  
const buttons = document.querySelectorAll('.arrow');
```

# Créer le tableau de slides dans lequel nous allons tourner

- je crée un index pour pouvoir comparer par la suite la slide en cours avec l'index pour les puces (*activer la colorisation de la puce en cours via une classe*)
- je crée la source de l'image
- le titre qui sera ajouté sur l'image
- et j'initialise le tableau à l'index 0

```
let slides = [  
  {  
    index: 0,  
    imageURL: 'slide1.jpg',  
    titleTag: 'Impressions tous formats <span>en boutique et en ligne</span>',  
  },  
  {  
    index: 1,  
    imageURL: 'slide2.jpg',  
    titleTag: 'Tirages haute définition grand format <span>pour vos bureaux et events</span>',  
  },  
  {  
    index: 2,  
    imageURL: 'slide3.jpg',  
    titleTag: 'Grand choix de couleurs <span>de CMJN aux pantones</span>',  
  },  
  {  
    index: 3,  
    imageURL: 'slide4.png',  
    titleTag: 'Autocollants <span>avec découpe laser sur mesure</span>',  
  },  
]
```

# Coté HTML

```
<div id="banner">
  
  <img id="slider_img" class="banner-img" src=""
  alt="Banner Print-it">
  <p id="p"></p>
  
  <div class="dots">
    <ul class='dots'>
      <li class="dot dot_selected"></li>
      <li class="dot"></li>
      <li class="dot"></li>
      <li class="dot"></li>
    </ul>
  </div>
</div>
```

Ici dans la bannière , une image qui a la classe arrow et arrow left pour la flèche gauche et une image qui a la classe arrow et arrow right pour la fleche droite.

Il y a l'image de la bannière qui a l'id slider\_img et la classe banner\_img la source sera ajoutée dynamiquement en js.

Il y a le p la ou sera injecté le titleTag

Il y a les dots créés avec le dot selected de depart qui corresponde au slideIndex de départ c' est à dire l'index 0 sur lequel j'ai défini.

# Créer la fonction *display()*

Je commence par créer la variable `titleTag` afin de récupérer le titre à afficher et ainsi avec `innerHTML` je l'affiche dans le paragraphe.

Ce **titleTag** prendra l'index en cours.

(ex: objet 0 => 0) => qui sera le *slideIndex* correspondant à 0.

Ensuite j'affiche la source de l'élément sur le même procédé en récupérant le nom de l'image (*slide1.png, slide2.png...*)

et ensuite je crée pour chaque puce elles auront un élément et un index si l'index du tableau correspond au *slideIndex* qui est l'index courant alors la classe sera ajoutée à l'élément en cours et retirée à tout autre également qui aurait pu avoir la classe `dot_selected`.

La fonction `display()` sera appelée une première fois afin de l'afficher puis lors du clic des boutons.

```
function display(){
  ... let titleTag = slides[slideIndex].titleTag;
  ... p.innerHTML = titleTag;
  ... slider_img.src = `./assets/images/slideshow/${slides
    [slideIndex].imageUrl}`;
  ... puces.forEach((element, index) => {
    ... if(index === slideIndex){
    ...   element.classList.add('dot_selected');
    ... }
    ... } else {
    ...   element.classList.remove('dot_selected');
    ... }
  ... });
};

display();
```



## *function* **beforeSlide()**

```
function beforeSlide(){  
  ...  
  if(slideIndex === 0){  
    slideIndex = slides.length - 1;  
  }  
  else{  
    slideIndex--;  
  }  
}
```

Ici la fonction ***beforeSlide()*** vérifie si le *slideIndex* est à 0 si c'est le cas alors *slideIndex* sera remis au dernier index du tableau afin d'afficher le dernier objet.  
Sinon *slideIndex* sera décrémenté de 1 à chaque fois qu'il sera appelé.


## *function beforeSlide()*

```
function nextSlide(){  
  ...  
  if(slideIndex === slides.length-1){  
    ... slideIndex = 0;  
  }  
  else{  
    ... slideIndex++;  
  }  
}
```

Ici la fonction ***nextSlide()*** vérifie si le *slideIndex* est à la fin du tableau si c'est le cas alors *slideIndex* sera remis au index 0 du tableau afin d'afficher le premier objet.  
Sinon *slideIndex* sera incrémenté de 1 à chaque fois qu'il sera appelé.



## La gestion des événements au clic

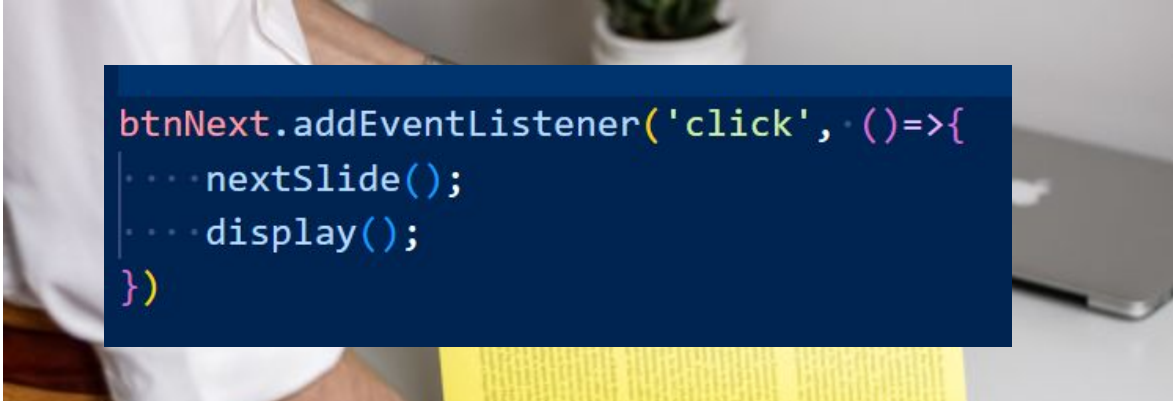


```
btnBefore.addEventListener('click',()=>{  
  ...beforeSlide();  
  ...display();  
})
```

Ici nous créons un événement sur le bouton *btnBefore* au clic de ce bouton , nous appelons la fonction *beforeSlide()* et affichons le display en conséquence que nous avons vu précédemment.

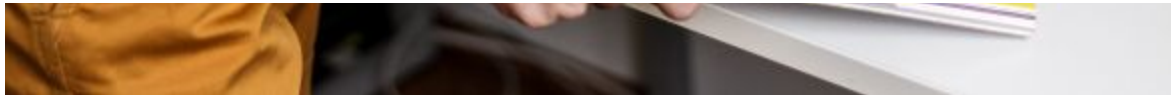


## La gestion des événements au clic



```
btnNext.addEventListener('click', ()=>{  
  ...nextSlide();  
  ...display();  
})
```

Ici nous créons un événement sur le bouton *btnNext* au clic de ce bouton , nous appelons la fonction ***nextSlide()*** et affichons le display en conséquence que nous avons vu précédemment.





Passons à la démonstration !