Tạo hiệu ứng slide tự động

1. Phân tích:
2. Thiết kế HTML
3. Định dạng CSS
4. Xây dựng JS
5. Định dạng css cho class phát sinh

sau khi đã phân tích các đối tượng và đã chuẩn bị xong các nguyên vật liệu, chuẩn bị làm việc nhé!

Bước 1: Tạo và thiết lập hiệu ứng các nút chuyển

-HTML:

```
<div class="nutchuyen">
    <ul>
        <li>1</li>
        <li>2</li>
        <li>3</li>
    </ul>
</div>
```

- CSS:

```
.nutchuyen{
    text-align: center;
    bottom: 1%;
    left: 50%;
    transform: translateX(-50%);
}
```

```
.nutchuyen ul li{
    display: inline-block;
    border: 3px solid white;
    width: 20px;
    height: 20px;
    border-radius: 50%;
    margin: 20px 10px;
    transition: 0.5s;
    cursor: pointer;
}
```

⇨ Xem kết quả.

Đinh dạng cho sự kiện khi click, rê chuột vào nút

```
.nutchuyen ul li:hover,
.nutchuyen ul li.active
{
    background: rgb(22, 22, 22);
}
```

⇨ Test kết quả

- Viết JS để xử lý khi click vào nut (lưu ý kiểm tra từng lịnh nhé)

```
document.addEventListener("DOMContentLoaded",function(){
    //khai báo đối tượng
    var nut = document.querySelectorAll('.nutchuyen ul li');
    // console.log(nut);
})
```

- Bắt sự kiện khi click vào nút

```
for (var i = 0; i < nut.length; i++) {
      nut[i].addEventListener('click',function(){
          // console.log(this);
          }
}
```

Chuyển qua html, thêm class active cho li đầu tiên: `<li class="active">1</li>`

Test sự kiện

Sau khi đã bắt được sự kiện, thêm class vào sự kiện

```
for (var i = 0; i < nut.length; i++) {
      nut[i].addEventListener('click',function(){
          // console.log(this);
           this.classList.add('active');
          }
}
```

Test đi nhé,

kết quả không như mong đợi, vì trước khi add class, chúng ta phải bỏ class tương ứng cho những đối tượng khác, sửa lại như sau:

```
for (var i = 0; i < nut.length; i++) {
        nut[i].addEventListener('click',function(){
            // console.log(this);
            for (var i = 0; i < nut.length; i++) {
                nut[i].classList.remove('active');
            }
            this.classList.add('active');
       }
}
```

Hoàn thành rồi ! ☺ ☺ ☺

Bước 2: xử lý slide

1. Thiết kế HTML:
   Chèn bên ngoài, phía trên nút chuyển

```
<div class="ndslide">
    <ul>
        <li class="active">…
        </li>
        <li>…
        </li>
        <li>…
        </li>
    </ul>
</div> <!--end slide-->
```

Trong mỗi <li> chèn các đối tượng sau:

```
<div class="slide">
        <div class="hinh" style="background-
image: url(images/slide1.jpg);"> </div>
        <img src="images/tg1.png" alt="" class="tgduoi1">
        <img src="images/tg1.png" alt="" class="tgduoi2">
        <img src="images/tg1_1.png" alt="" class="tgtren">
        <div class="noidung">
                <h2>01</h2>
                <small>Charlie</small>
                <p>Per inceptos himenaeos. Mauris in erat justo.
 Nullam ac urna eu felis</p>
                <div class="nut">Sea the project</div>
         </div>
    </div>
```

Thay đổi nội dung trong các thẻ cho phù hợp

⇨ Test xem thế nào => không như mong đợi

2. Xử CSS cho hình hiện ra và full mà hình

- Vì các đối tượng nằm chồng lên nhau nên bắt buộc phải có vị trí, và kích thước phải full màn hình:

```
.ndslide ul li{
    list-style: none;
    position: absolute;
    width: 100%;
    height: 100%;
}
```

```
.slide .hinh{
    width: 100%;
    height: 100%;
    background-size: cover;
}
```

⇨ **Test**

Để hình hiện ra và full, bắt buộc các thẻ cha phai có kích thước

```
html,body{width: 100%;height: 100%;}
.ndslide{height: 100%; width: 100%;}
.ndslide ul{ position: relative; height: 100%; width: 100%;}
```

- Thêm thuộc tính: `overflow-x: hidden;` cho html,boydy để ẩn thanh cuộn ngang

3. Xử lý hiển thị cho 1 slide (đối tượng slide là cha, chứa các đối tượng bên trong)

```
.ndslide ul li .slide{
    position: relative;
}
.slide .tgduoi1,
.slide .tgduoi2,
.slide .tgtren,
.slide .noidung{
    position: absolute;
    z-index: 1;
    top: 0;
}
```

⇨ **Kiểm tra**

- Điều chỉnh cho từng đối tượng
- Cho các đối tượng ẩn để dễ kiểm tra, `opacity: 0;`

Bắt đầu định dạng từng đối tượng

```
.slide .tgduoi1{ opacity: 1;top:-50% ;}
.slide .tgduoi2{opacity: 1; top:-20%;}
.slide .tgtren{opacity: 1;right: -5%;}
.slide .noidung{ width: 20%;left: 10%;
    z-index: 3; opacity: 1;top: 46%; color: wheat;}
```

Điều chỉnh cho các hình tam giác : `opacity: 0;`

- Cho slide được kích hoạt hiện ra:

```
.ndslide li.active{opacity: 1;}
```

và thêm thuộc tính chuyển động chậm cho slide

```
.ndslide ul li{transition: 0.5s;}
```

- Xử lý cho các nút hiện lên trên slide

Bổ sung code sau:

Thêm 1 div chứa toàn bộ slide và nút, đặt div này là cha

```
<div class="slidewrapper">

    <div class="ndslide">…
    </div> <!--end slide-->

    <div class="nutchuyen">…
    </div>
</div> <!-- end slie wrapper -->
```

CSS: `slidewrapper{ position: relative; height: 100%; width: 100%;`

`overflow: hidden;}`

bổ sung thuộc tính sau vào class nutchuyen

```
.nutchuyen{position: absolute;z-index: 5;}
```

- Tiếp tục css cho các nội dung text

```
.slide small{
    font-size: 30px;
    font-weight: bold;
    padding: 10px 0;
    display: block;
    text-decoration: underline;
}
.slide h2{font-size: 80px; font-weight: normal;}
.noidung .nut{
     border: 1px solid white;
     width: 55%;
     padding: 8px 15px;
     text-decoration: none;}
```

kiểm tra vị trí của nút được click (JS)

```
for (var i = 0; i < nut.length; i++) {
        nut[i].addEventListener('click',function(){
            for (var i = 0; i < nut.length; i++) {
                nut[i].classList.remove('active');
            }
            this.classList.add('active');
            //xử lý tính vị trí
            // console.log(this.previousElementSibling);
```

Nhập và consol, click vào từng nút xem kết quả để hiểu vấn đề

- Xử lý cho nút được kích hoạt (thay code // bằng nội dung sau)

```
var nutactive=this;
var vt=0;
for (vt = 0; nutactive=nutactive.previousElementSibling; vt++)
{ }
```

// nutactive=nutactive.previousElementSibling: cho nút kích hoạt bằng nút trước nó

giải thích: vòng lặp không làm gì cả, mục đích để lần ra biến vt

kiểm tra lại:

```
var nutactive=this;
var vt=0;
for (vt = 0; nutactive=nutactive.previousElementSibling; vt++)
{ }
 // console.log('vi tri nut kich hoat ='+vt);
```

Xử dụng console.log kiểm tra vị trí nút được click

Trước khi viết tiếp xử lý cho slide được hiển thị khi click vào nút, các bạn thêm class sau vào css

```
.ndslide li.active{opacity: 1;}
```

- Chuyển qua JS, thêm code
  Khai báo biến để lấy đối tượng slide

```
var slides = document.querySelectorAll('.ndslide ul li');
// cho tất cả slide ẩn đi và hiện slide được clcik
    for (var i = 0; i < slides.length; i++) {
            slides[i].classList.remove('active');
            slides[vt].classList.add('active');
    }
```

Các bạn test kết quả

>>

Code tham khảo

```
document.addEventListener("DOMContentLoaded",function(){
    var nut = document.querySelectorAll('.nutchuyen ul li');
    var slides = document.querySelectorAll('.ndslide ul li');
    for (var i = 0; i < nut.length; i++) {
        nut[i].addEventListener('click',function(){
            for (var i = 0; i < nut.length; i++) {
                nut[i].classList.remove('active');
            }
            this.classList.add('active');
            var nutactive=this;
            var vt=0;
           for (vt = 0; nutactive=nutactive.previousElementSibling; vt++)
            {   }
           for (var i = 0; i < slides.length; i++) {
                slides[i].classList.remove('active');
                slides[vt].classList.add('active');
            }
        })
    }  //end sự kiện nut
})
```

Bước 3: Xử lý cho hiệu ứng tự chuyển động (kết hợp animation)

- Khai báo hàm auto

```
document.addEventListener("DOMContentLoaded",function(){
.........
} //end sự kiện nut
    autoslide();
    //hàm tự chuyển slide
    function autoslide(){
    }
})
```

- Xây dựng hàm lấy thời gian để chạy slide

```
function autoslide(){
            //lấy thời gian
    var thoigian = setInterval(function(){},1000)
```

dùng console kiểm tra kết quả

- Xử lý cho các slide được kích hoạt khi click

```
var thoigian = setInterval(function(){
        var vtslide=0;
        var slidehientai=document.querySelector('.ndslide ul li.a
ctive');
```

```
            for (var vtslide = 0; slidehientai=slidehientai.previousE
   lementSibling; vtslide++) {   }
            if(vtslide<(slides.length-1)){
                for (var i = 0; i < slides.length; i++) {
                    slides[i].classList.remove('active');
                    nut[i].classList.remove('active');
                }
                  //cho phan tu tiep theo của slide hiên ra
      slides[vtslide].nextElementSibling.classList.add('active');
      nut[vtslide].nextElementSibling.classList.add('active');
            }
            else //ẩn các phan tu khác
            {
                for (var i = 0; i < slides.length; i++) {
                    slides[i].classList.remove('active');
                    nut[i].classList.remove('active');
                }
            //cho phan tu tiep theo của slide hiên ra
            slides[0].classList.add('active');
            nut[0].classList.add('active');
            }
           },6000);
```

- CSS cho các đối tượng tự hiện ra (animation)

```
ul li.active .slide .tgduoi1{
    animation: hientgduoi1 2s forwards;}
@keyframes hientgduoi1{
    from{transform: translateX(-
100%) translateY(100%); opacity: 0;}
    to{transform: translateX(0) translateY(0); opacity: 0.2;}
}
ul li.active .slide .tgduoi2{
    animation: hientgduoi2 2s forwards;
    animation-delay: 0.2s;
}
@keyframes hientgduoi2{
    from{transform: translateX(-
100%) translateY(100%); opacity: 0;}
    to{transform: translateX(0) translateY(0); opacity: 0.5;}
}
```

```
ul li.active .slide .tgtren{
    animation: hientgtren 2s forwards;
    /* animation-delay: 0.2s; */
}
@keyframes hientgtren{
    from{transform: translateX(100%) translateY(-
100%); opacity: 0;}
     to{transform: translateX(0) translateY(0); opacity: 0.2;}
}

ul li.active .noidung h2,
ul li.active .noidung small,
ul li.active .noidung p,
ul li.active .noidung .nut{
    opacity: 0;
    animation: hiennd 3s forwards;
}
@keyframes hiennd{
    from{transform: translateX(-30%); opacity: 0;}
    to{transform: translateX(0); opacity: 1;}
}
ul li.active .noidung h2 {animation-delay: 0.8s}
ul li.active .noidung small{animation-delay: 1s}
ul li.active .noidung p{animation-delay: 1.2s}
ul li.active .noidung .nut{animation-delay: 1.4s}
```

TEST KẾT QUẢ => QUÁ OK

>>