HỌC PHẦN: ĐỒ HỌA ỨNG DỤNG

GAME BRICK OUT SỬ DỤNG PIXIJS

Nhóm 3

Thành viên:

Vũ Hồng Phúc
18120515

Lê Trọng Việt
19120716

Phần 1

- PixiJS

Phần 2

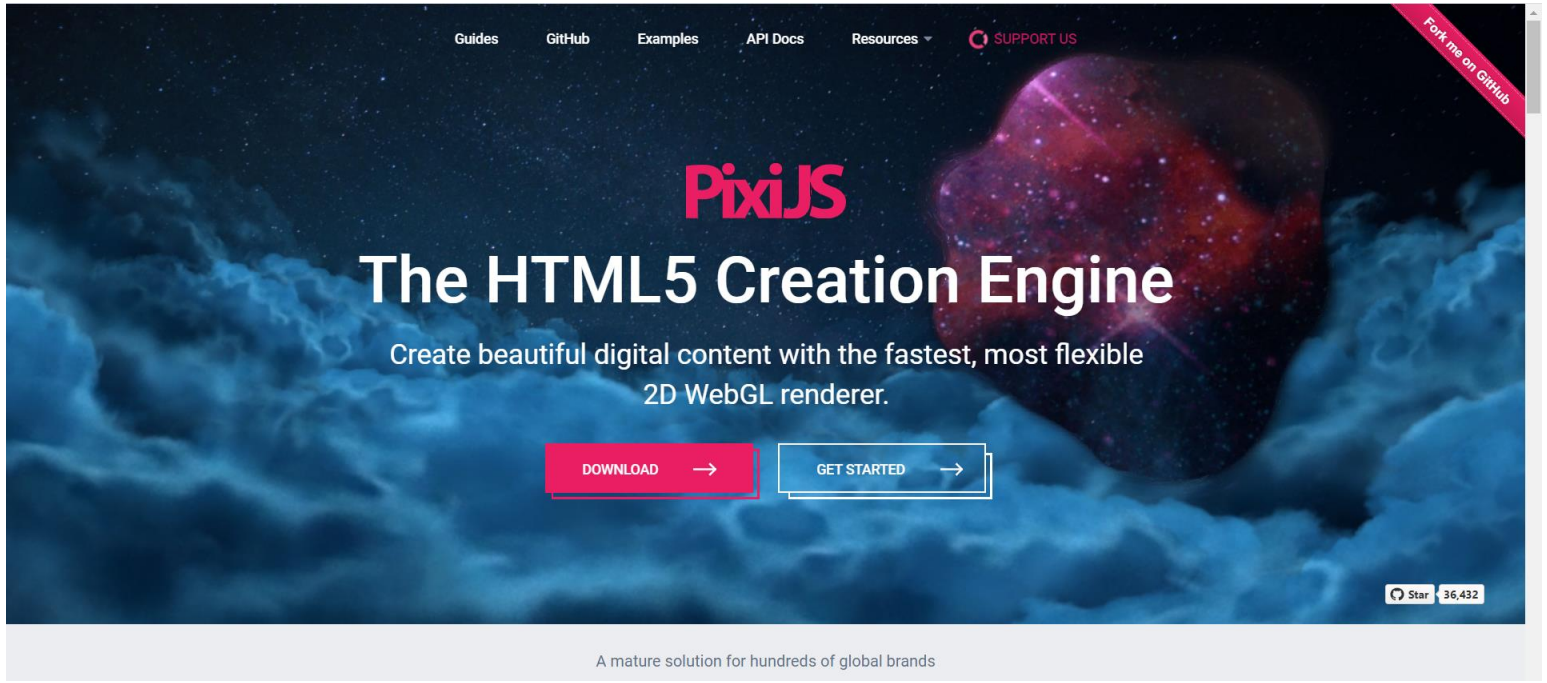
- Game Brick Out

Phần 1

- PixiJS

Tổng quát PiciJS

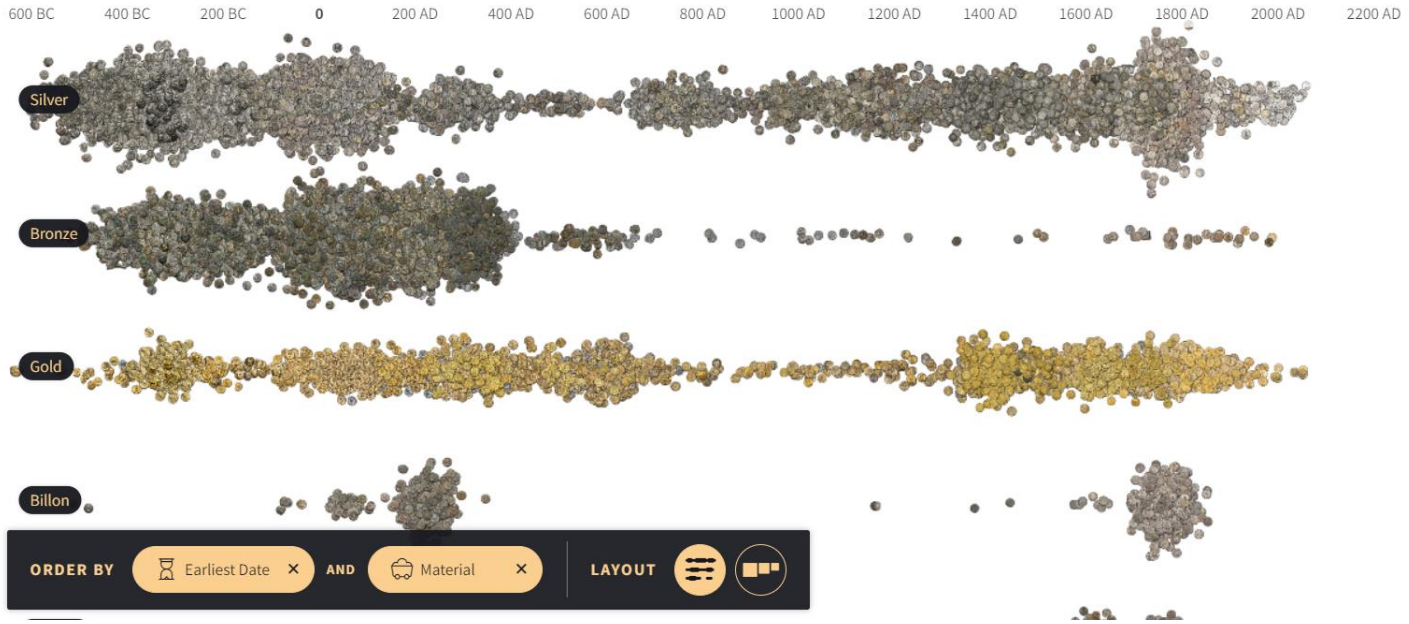
- PixiJS là một công cụ hiển thị trên nền tảng web, cho phép sử dụng sức mạnh của WebGL hoặc canvas để render các nội dung 2D trên màn hình một cách liên tục



Giao diện trang web PixiJS

Tổng quát PiciJS

- Cung cấp hỗ trợ tương tác để xử lý các click và touch events
- Thường được sử dụng trong làm game, trực quan hóa dữ liệu, giáo dục, …
- PiciJS cũng có thể sử dụng cho ứng dụng desktop hoặc mobile



Lịch sử các đồng xu được sắp xếp theo thuộc tính và trực quan hóa sử dụng PixiJS

Các thành phần chính

Một số thành phần chính của PiciJS

- **Renderer:**

  - Vẽ biểu đồ cảnh và tất cả các nội dung lên canvas sử dụng WebGL

  - Gồm nhiều thành phần quản lý cụ thể, các thành phần được thêm mặc định khi tạo Renderer như PIXI.ContextSystem, PIXI.EventSystem, …

  - Tạo một renderer với cú pháp:

RendererName = new PIXI.Renderer(Options)

Với các options là các tham số truyền vào như options.width, options.hight, options.view, …

  - Hỗ trợ nhiều phương thức như **registerPlugin, clear, render,** ...

Các thành phần chính

Một số thành phần chính của PiciJS

- **Container:**

  - Một thùng chứa để hiển thị tất cả những đối tượng con bên trong nó, ví dụ như Graphics, Sprite, …
  - Chứa các thuộc tính như: **children, height, interactiveChildren, sortableChildren, sortDirty, width**
  - Các phương thức của một container như **addChild**(children), **addChildAt**(child,index), **getChildAt**(index), **render**(renderer), …

Các thành phần chính

Một số thành phần chính của PiciJS

- **Loader:**
  - Cung cấp công cụ để tải tài nguyên lên như hình ảnh hay âm thanh
  - Tạo một Loader mới:

    loaderName = new PIXI.Loader (baseUrl, concurrency)

    Với baseUrl là url cơ sở dẫn tới tài nguyên, concurrency là số tài nguyên được tải lên đồng thời

  - Thêm tài nguyên vào loader, sử dụng **add**
  - Để load tài nguyên lên sử dụng phương thức **load()**

```
loader
    // normal param syntax
    .add('key', 'http://...', function () {})
    .add('http://...', function () {})
    .add('http://...')
```

Các thành phần chính

Một số thành phần chính của PiciJS

- **Application:** Gom Loader, Ticker, Renderer thành một để dễ làm việc

- Tạo Application: appName = **new PIXI.Application** (options)
  Với options là các tham số cho renderer

- Có các thành phần như **stage** (container chứa các đối tượng), **ticker** (cập nhật các đối tượng), …

- Có các phương thức hay được sử dụng như **destroy(), render(), start(), stop(), …**

Các thành phần chính

Một số thành phần chính của PiciJS

- **Interaction:** Hỗ trợ cả touch và mouse click
- **Ticker:** Cập nhật và xử lý trạng thái của các đối tượng qua các vòng lặp
- **Sprite:** Đối tượng đơn giản và phổ biến nhất trong PixiJS, gồm một hình ảnh được render lên màn hình ví dụ như các con quái trong một trò chơi
- **Graphic:** Làm việc với các đối tượng đồ họa

Các thành phần chính

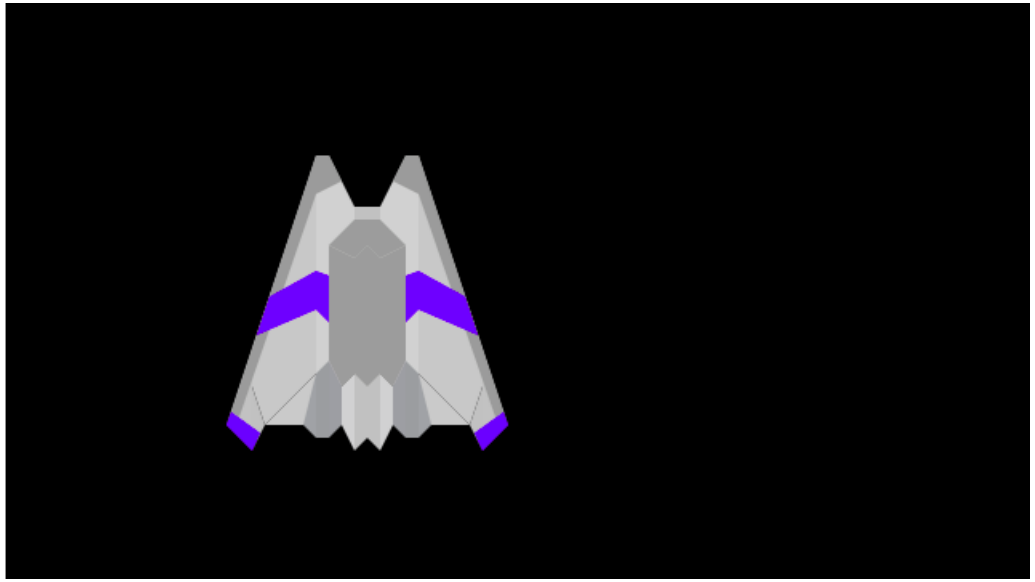
```
<!doctype html>
<html>
  <head>
    <script src="https://pixijs.download/release/pixi.min.js"></script>
  </head>
  <body>
    <script>
      // Create the application helper and add its render target to the page
      let app = new PIXI.Application({ width: 640, height: 360 });
      document.body.appendChild(app.view);

      // Create the sprite and add it to the stage
      let sprite = PIXI.Sprite.from('sample.png');
      app.stage.addChild(sprite);

      // Add a ticker callback to move the sprite back and forth
      let elapsed = 0.0;
      app.ticker.add((delta) => {
        elapsed += delta;
        sprite.x = 100.0 + Math.cos(elapsed/50.0) * 100.0;
      });
    </script>
  </body>
</html>
```

Đoạn code đơn giản sử dụng PixiJS

Các thành phần chính



Kết quả
(Tàu vũ trụ di chuyển liên tục từ trái sang phải và ngược lại)

Phần 2

- Game Brick Out

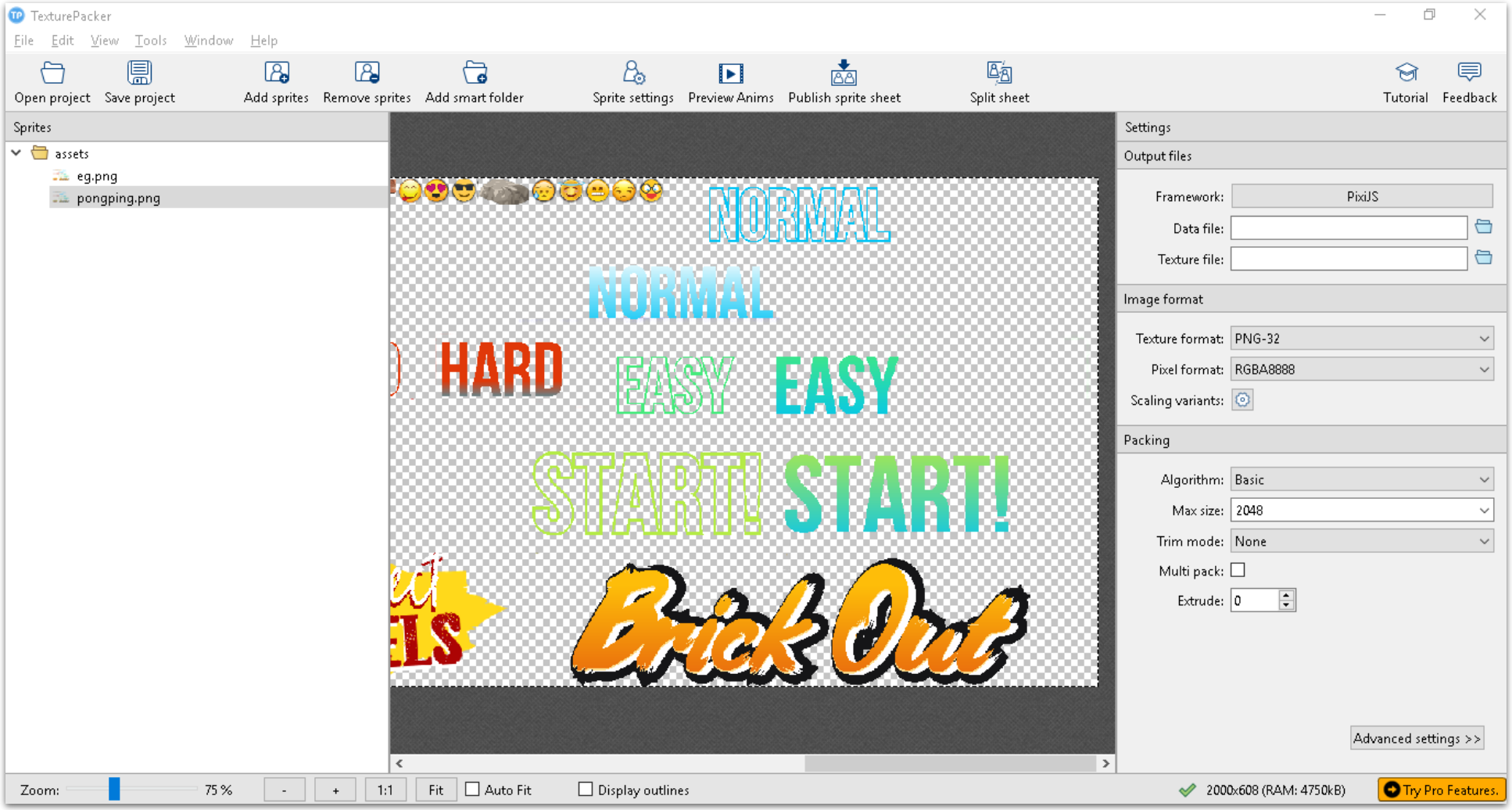
Spirte trong game 2D

- **Sprite** là ảnh 2 chiều của một đối tượng ở trạng thái nhất định.
- Spirte được dùng để tạo animation cho 2D game, website.

16 sprites

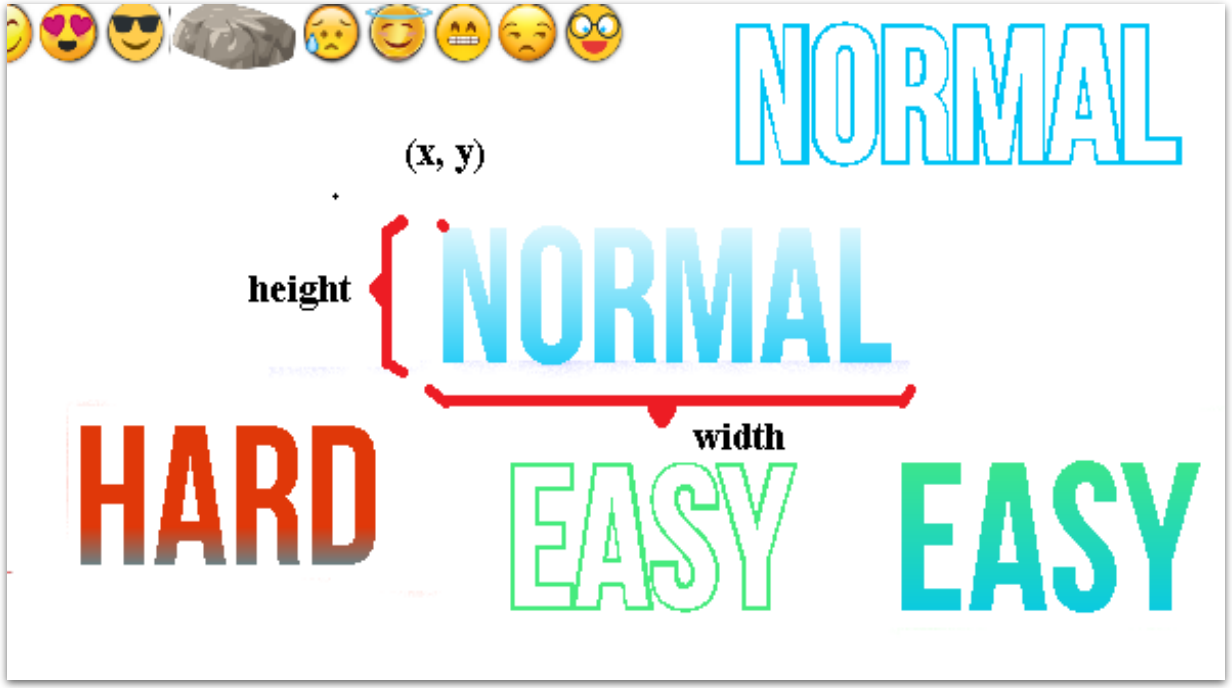
Tạo spirte sheet

- **Spirte sheet** là tập hợp của nhiều spirte, bao gồm thông tin định vị mỗi spirte trong đó.
- Có thể tạo spirte sheet bằng phần mềm **TexturePacker.**



Tạo spirte sheet

- **TexturePacker** hỗ trợ ghép nhiều spirte lại với nhau và kết xuất thông tin tọa độ của chúng vào file **json.**



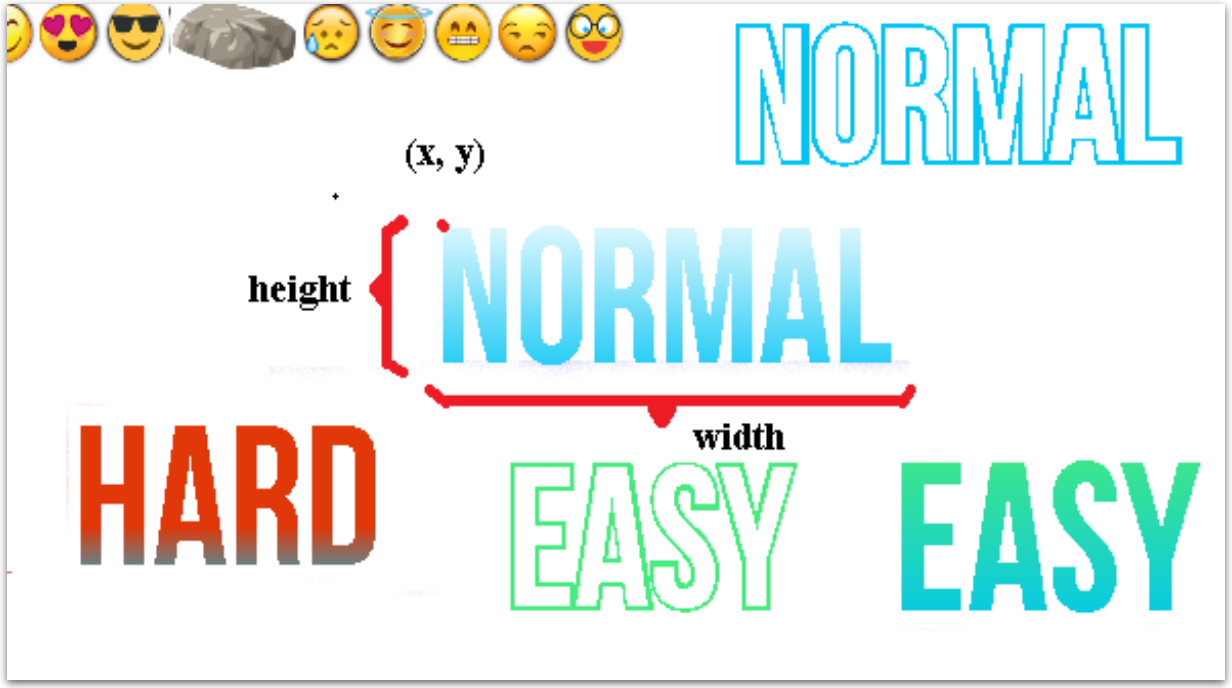
File ảnh các spirte

```
"normal-default.png":
{
    "frame": {"x":482,"y":0,"w":350,"h":90},
    "rotated": false,
    "trimmed": false,
    "spriteSourceSize": {"x":0,"y":0,"w":350,"h":90},
    "sourceSize": {"w":350,"h":90}
},
"normal-hover.png":
{
    "frame": {"x":350,"y":90,"w":350,"h":90},
    "rotated": false,
    "trimmed": false,
    "spriteSourceSize": {"x":0,"y":0,"w":350,"h":90},
    "sourceSize": {"w":350,"h":90}
},
"play-clicked.png":
{
    "frame": {"x":0,"y":302,"w":300,"h":150},
```

File json chứa thông tin các spirte

Tạo spirte sheet

- **TexturePacker** hỗ trợ ghép nhiều spirte lại với nhau và kết xuất thông tin tọa độ của chúng vào file **json.**



File ảnh các spirte

```
"normal-default.png":
{
    "frame": {"x":482,"y":0,"w":350,"h":90},
    "rotated": false,
    "trimmed": false,
    "spriteSourceSize": {"x":0,"y":0,"w":350,"h":90},
    "sourceSize": {"w":350,"h":90}
},
"normal-hover.png":
{
    "frame": {"x":350,"y":90,"w":350,"h":90},
    "rotated": false,
    "trimmed": false,
    "spriteSourceSize": {"x":0,"y":0,"w":350,"h":90},
    "sourceSize": {"w":350,"h":90}
},
"play-clicked.png":
{
    "frame": {"x":0,"y":302,"w":300,"h":150},
```

File json chứa thông tin các spirte

Thư viện Bump – Xử lý va chạm cho PixiJS

- **Bump** là thư viện mở bao gồm các hàm hỗ trợ xử lý va chạm vật lý cho PixiJS (phiên bản stable 3.0.11). Github: bump.js.
- Một số hàm trong Bump bao gồm:
  + Nhóm hàm **Hit:** kiểm tra các đối tượng có va chạm với nhau hay không.
  + Nhóm hàm **Collision:** tính toán và tạo chuyển động nảy khi hai đối tượng va chạm.