loadXML()

Loads an XML file to create a p5.XML object.

Extensible Markup Language (XML) is a standard format for sending data between applications. Like HTML, the XML format is based on tags and attributes, as in `<time units="s">1234</time>`.

The first parameter, `path`, is always a string with the path to the file. Paths to local files should be relative, as in `loadXML('/assets/data.xml')`. URLs such as `'https://example.com/data.xml'` may be blocked due to browser security.

The second parameter, `successCallback`, is optional. If a function is passed, as in `loadXML('/assets/data.xml', handleData)`, then the `handleData()` function will be called once the data loads. The p5.XML object created from the data will be passed to `handleData()` as its only argument.

The third parameter, `failureCallback`, is also optional. If a function is passed, as in `loadXML('/assets/data.xml', handleData, handleFailure)`, then the `handleFailure()` function will be called if an error occurs while loading. The `Error` object will be passed to `handleFailure()` as its only argument.

Note: Data can take time to load. Calling `loadXML()` within preload() ensures data loads before it's used in setup() or draw().

Examples


```
let myXML;

// Load the XML and create a p5.XML object.
function preload() {
  myXML = loadXML('/assets/animals.xml');
}

function setup() {
  createCanvas(100, 100);

  background(200);

  // Get an array with all mammal tags.
  let mammals = myXML.getChildren('mammal');

  // Style the text.
  textAlign(LEFT, CENTER);
  textFont('Courier New');
  textSize(14);

  // Iterate over the mammals array.
  for (let i = 0; i < mammals.length; i += 1) {

    // Calculate the y-coordinate.
    let y = (i + 1) * 25;

    // Get the mammal's common name.
    let name = mammals[i].getContent();

    // Display the mammal's name.
    text(name, 20, y);
```

```
let lastMammal;

// Load the XML and create a p5.XML object.
function preload() {
  loadXML('/assets/animals.xml', handleData);
}

function setup() {
  createCanvas(100, 100);

  background(200);

  // Style the text.
  textAlign(CENTER, CENTER);
  textFont('Courier New');
  textSize(16);

  // Display the content of the last mammal element.
  text(lastMammal, 50, 50);

  describe('The word "Zebra" written in black on a gray
background.');
}

// Get the content of the last mammal element.
function handleData(data) {
  // Get an array with all mammal elements.
  let mammals = data.getChildren('mammal');

  // Get the content of the last mammal.
```

```
let lastMammal;

// Load the XML and preprocess it.
function preload() {
  loadXML('/assets/animals.xml', handleData, handleError);
}

function setup() {
  createCanvas(100, 100);

  background(200);

  // Style the text.
  textAlign(CENTER, CENTER);
  textFont('Courier New');
  textSize(16);

  // Display the content of the last mammal element.
  text(lastMammal, 50, 50);

  describe('The word "Zebra" written in black on a gray
background.');
}

// Get the content of the last mammal element.
function handleData(data) {
  // Get an array with all mammal elements.
  let mammals = data.getChildren('mammal');

  // Get the content of the last mammal.
```

Syntax

```
loadXML(path, [successCallback], [errorCallback])
```

Parameters

path	String: path of the XML file to be loaded.
successCallback	Function: function to call once the data is loaded. Will be passed the p5.XML object.
errorCallback	Function: function to call if the data fails to load. Will be passed an `Error` event object.

Returns

p5.XML: XML data loaded into a p5.XML object.

This page is generated from the comments in src/io/files.js . Please feel free to edit it and submit a pull request!

Related References

addChild
Adds a new child element and returns a reference to it.

getAttributeCount
Returns the number of attributes the element has.

getChild
Returns the first matching child element as a new p5.XML object.

getChildren
Returns an array with the element's child elements as new p5.XML objects.