



POLITECNICO
MILANO 1863

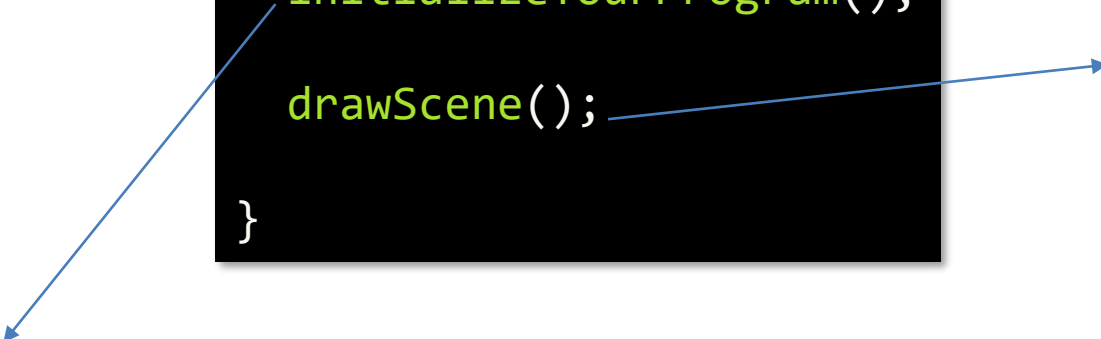
Cleaning the code

Computer Graphics 2020

Erica Stella (erica.stella@polimi.it)

Your WebGL Program

```
function main() {  
    getCanvas();  
    initializeYourProgram();  
    drawScene();  
}
```



```
function initializeYourProgram(){  
    compileAndLinkShaders();  
    getAttributesAndUniformLocations();  
    createVAO();  
    putAttributesOnGPU();  
}
```

```
function updateTransformationMatrices()  
{  
    updateModel();  
    updateView();  
    updatePerspective();  
}  
  
function drawScene() {  
    updateTransformationMatrices();  
    bindVertexArray();  
    sendUniformsToGPU();  
    drawElements()/drawArray();  
  
    window.requestAnimationFrame(drawScene)  
    ;  
}
```

Cleanup the code

In the previous lectures we defined vertex and fragment shaders as hard-coded strings

```
var vertexShaderSource = `#version 300 es
    in vec3 a_position;
    uniform mat4 matrix;
    void main() {
        gl_Position = matrix * vec4(a_position,1.0);
    }`;
var fragmentShaderSource = `#version 300 es
    precision mediump float;
    out vec4 outColor;
    void main() {
        outColor = vec4(1.0,0.0,0.0,1.0);
    }`;
```

[...]

```
var vertexShader = utils.createShader(gl, gl.VERTEX_SHADER, vertexShaderSource);
var fragmentShader = utils.createShader(gl, gl.FRAGMENT_SHADER, fragmentShaderSource);
var program = utils.createProgram(gl, vertexShader, fragmentShader);
```

Cleanup the code

A cleaner approach is to store both vertex and fragment shaders into separate files...

```
#version 300 es

in vec3 a_position;

uniform mat4 matrix;

void main() {

    gl_Position = matrix * vec4(a_position,1.0);

}
```

vs.glsl

```
version 300 es

precision mediump float;

out vec4 outColor;

void main() {

    outColor = vec4(1.0,0.0,0.0,1.0);

}
```

fs.glsl

Cleanup the code

... and load `vs.glsl` and `fs.glsl` with javascript `fetch()` function (in the code of the `utils.loadFiles()` function provided in the `utils.js` file)

```
var path = window.location.pathname;
var page = path.split("/").pop();
var baseDir = window.location.href.replace(page, '');
var shaderDir = baseDir+"shaders/";

await utils.loadFiles([shaderDir + 'vs.glsl', shaderDir + 'fs.glsl'], function
(shaderText) {
    var vertexShader = utils.createShader(gl, gl.VERTEX_SHADER, shaderText[0]);
    var fragmentShader = utils.createShader(gl, gl.FRAGMENT_SHADER, shaderText[1]);
    program = utils.createProgram(gl, vertexShader, fragmentShader);

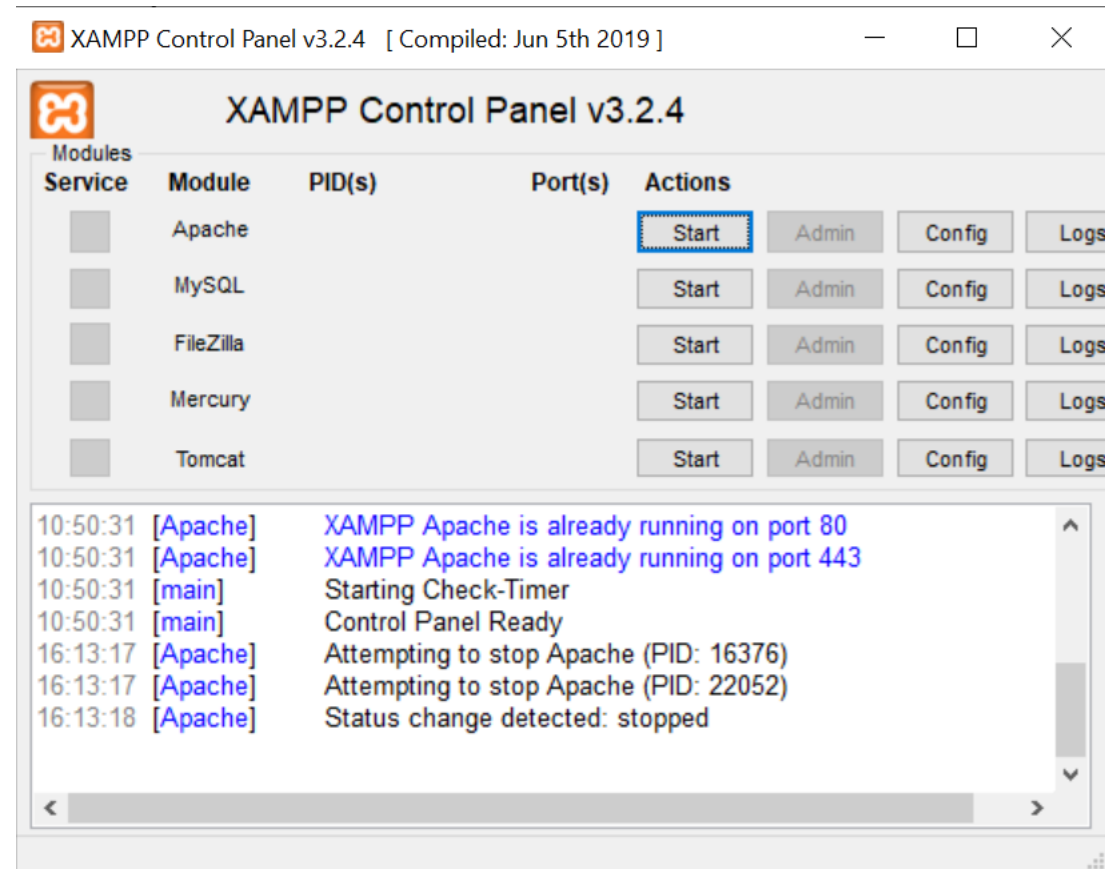
});
```

For security reasons the browsers stops this request. To make it work you have to setup a webServer

Setting the environment

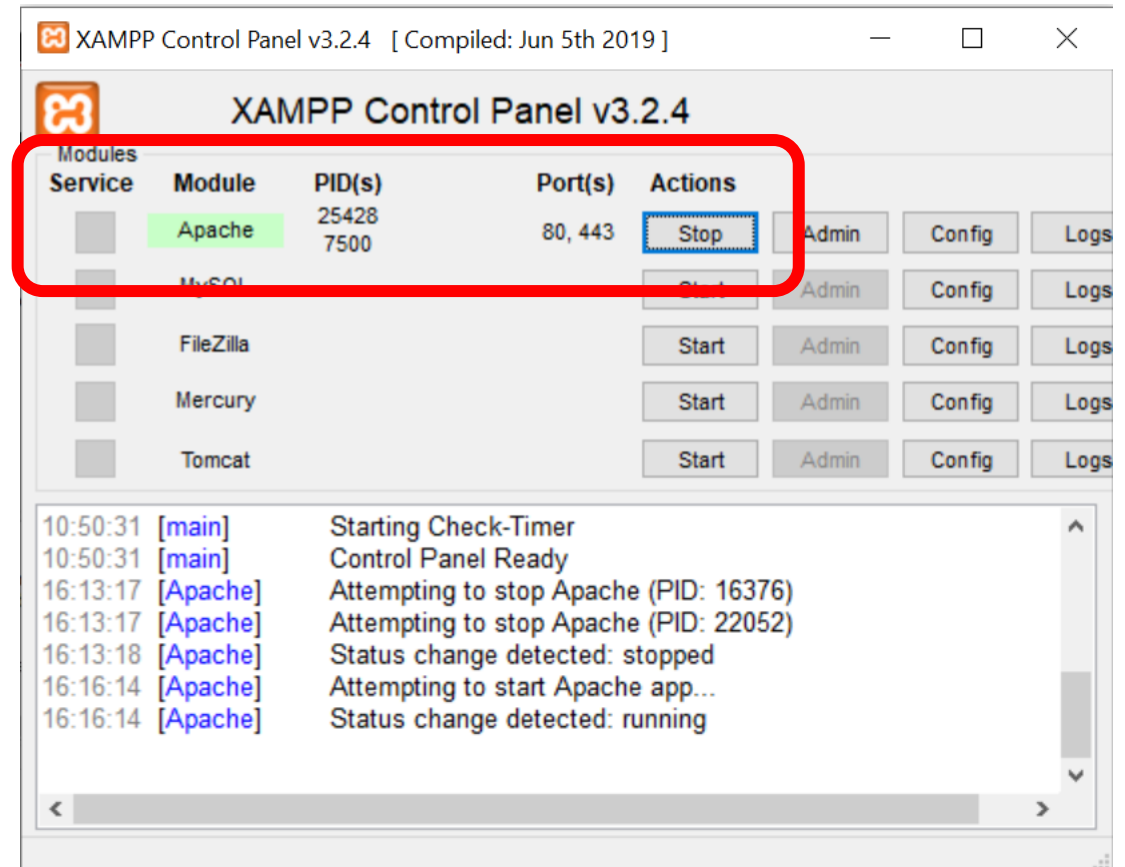
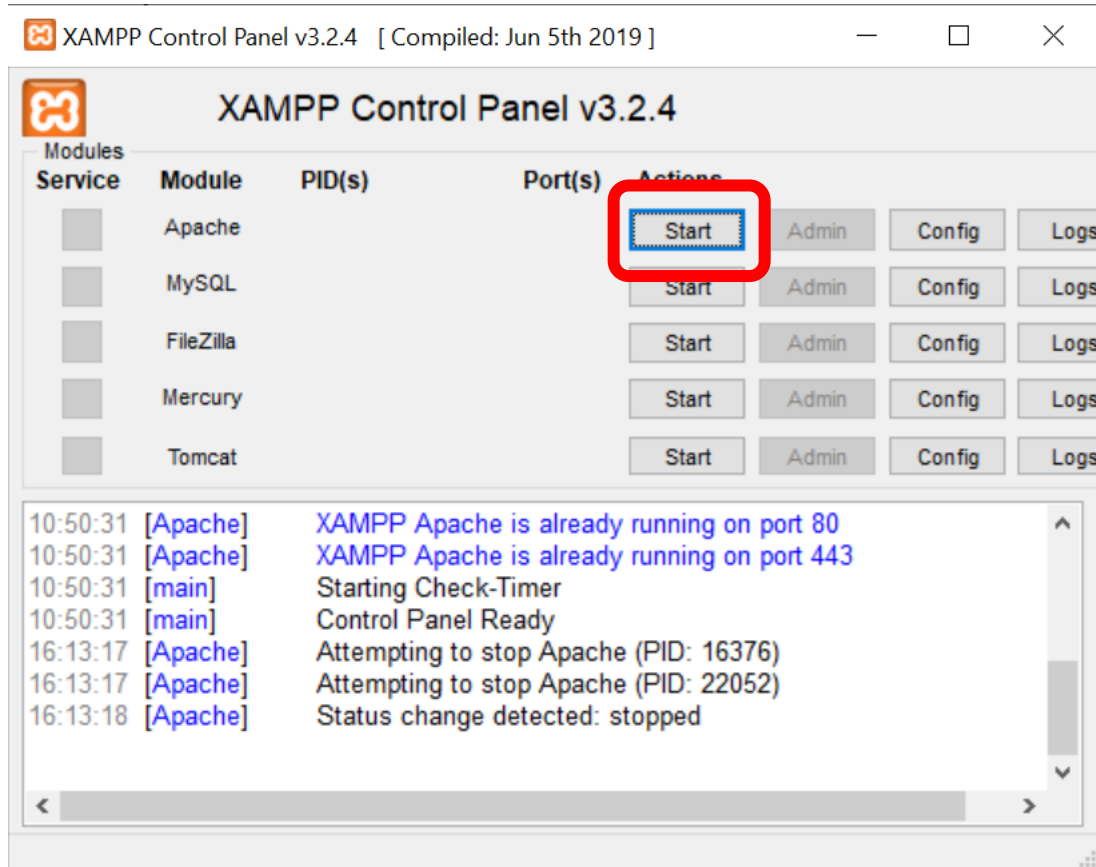
We propose here an easy way to set up a simple (and VERY unsafe) web server...

- Download XAMPP: <https://www.apachefriends.org/download.html>
- Install it
- Open *xampp-control-panel*



Setting the environment

- Press *Start* on Apache



Setting the environment

- Put all your files under the *htdocs* folder (~xampp/htdocs/)
- Now, when opening your html files, you **MUST** replace the path until *htdocs* with 127.0.0.1:

❯ file:///C:/Users/erica/Documents/xampp/htdocs/04-Texture/04a-TexturedCube/index.html



❯ 127.0.0.1/04-Texture/04a-TexturedCube/index.html