Pasantías realiabytics

**Capitulo 1**

Implementar de forma nativa y en angular, sin la ayuda de un tercero, el visualizador de Autodesk VIewer:

Pasos para iniciar Forge Viewer en angular:

1. Crear app angular
2. Instalar dependencias.

npm install --save-dev @types/three @types/forge-viewer

npm install --save-dev @types/forge-apis

1. // tsconfig.json **or** your framework-specific config file such **as** tsconfig.app.json
2. // en caso de angular tsconfig.app.json

{

    "types": ["forge-viewer"]

}

1. App.component.ts

import { Component, OnInit } from '@angular/core';

// import { MyAwesomeExtension } from './myawesomeextension'; extension

let urn:string= “urn here”;

@Component({

  selector: 'app-root',

  templateUrl: './app.component.html',

  styleUrls: ['./app.component.css']

})

export class AppComponent implements OnInit {

  public viewer!: Autodesk.Viewing.Viewer3D;

  async ngOnInit() {

    this.initViewer();

  }

  private async initViewer() {

    var viewer: Autodesk.Viewing.GuiViewer3D;

    let documentId:string = "urn:"+urn;

    var options = {

      env: 'AutodeskProduction',

      getAccessToken: (

        onGetAccessToken: (token: string, expire: number) => void

        ) => {

          fetch('/api/forge/oauth/token')

          .then((response) => response.json())

          .then((json) => {

          console.log(json);

              onGetAccessToken(json.access\_token,

              json.expires\_in);

            });

          },

         };

      Autodesk.Viewing.Initializer(options, function() {

        var htmlDiv: any = document.getElementById('forgeViewer');

        viewer = new Autodesk.Viewing.GuiViewer3D(htmlDiv,{ extensions: ['Autodesk.DocumentBrowser', “posible\_extencion” ]});

        var startedCode = viewer.start();

        if (startedCode > 0) {

        console.error('Failed to create a Viewer: WebGL not supported.');

        return;

      }

      console.log('Initialization complete, loading a model next...');

      Autodesk.Viewing.Document.load(documentId, onDocumentLoadSuccess, onDocumentLoadFailure);

    });

    function onDocumentLoadSuccess(viewerDocument:any) {

      // viewerDocument is an instance of Autodesk.Viewing.Document

      var defaultModel = viewerDocument.getRoot().getDefaultGeometry();

      viewer.loadDocumentNode(viewerDocument, defaultModel);

    }

    function onDocumentLoadFailure() {

      console.error('Failed fetching Forge manifest');

    }

  }

}

1. Index.html

Agregar al head

 <link rel="stylesheet" href="https://developer.api.autodesk.com/modelderivative/v2/viewers/7.0/style.min.css" type="text/css">

  <script src="https://developer.api.autodesk.com/modelderivative/v2/viewers/7.0/viewer3D.min.js"></script>

1. App.component.html

<div class="viewer" id="forgeViewer"></div>

1. App.component.css

.viewer {

  width: 100%;

  height: 100%;

  margin: 0;

  background-color: #F0F8FF;

}

**Capítulo 2**

Mejorar la velocidad de carga, disminuyendo la carga del visor

2.1 aumentar el límite de memoria preasignado y deshabilitar extensiones:

let config3d = {

      // loaderExtensions: { svf: 'Autodesk.MemoryLimited' }, // limita la memoria

      memory: {

       limit: 2050,

       debug :{

         force:true,

        }

        /\*  debug: {

          force:true,

          maxPageOutSize: 4050,                                // Max we will page out in one go

          // pixelCullingEnable: true,   // Useful with on demand loading

          // pixelCullingThreshold: true,

          occlusionThreshold: 4096,

          occlusionTestThreshold: 4096,

          startOcclusionTestingPackCount: 4096,

          testPackfileCount: 4,

          useOcclusionInstancing: true,

          automaticRefresh: true,

          boxProxyMaxCount: 2050, // show this many boxes during a render

          boxProxyMinScreen: 10 // if entire render batch is >= 1/10 of the scr

        }\*/

      },

/\*       disabledExtensions: { //deshabilita extensiones predeterminadas

        measure:true, //medir

        //  viewcube:true, //nada

        // layermanage:true, //nada

        explode:true, // separar

        section:true, //seccion analicis

        //  hyperlink:true, //nada

        bimwalk:true, //primera persona

        // fusionOrbit:true,

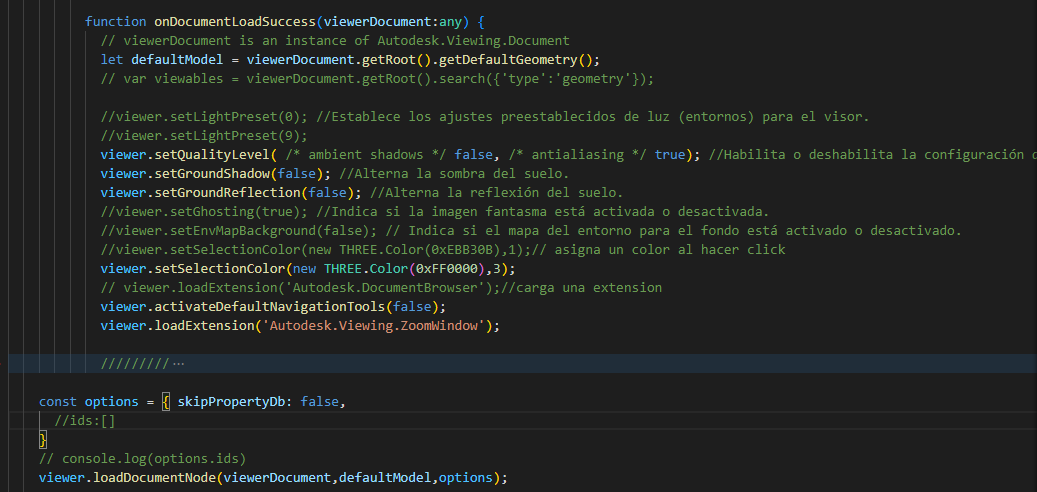
      }, \*/

      theme:'dark-theme',

      extensions: [LoggerExtension.extensionName, ],

    }

2.2 deshabilitar opciones de configuración predeterminadas, y deshabilitar propiedades del modelo y cargar una parte del modelo con su ids



**Capítulo 3**

**Reality capture**

(todas las solicitudes fueron realizadas en postman)

**paso 1 crear foto escena**

notas:

* el token de acceso debe tener permisos de whrite
* scenename tiene que ser un nombre unico
* obj es el formato admitido probado que se le puede hacer derivativa (cargar al viewer)
* otros formatos rcm, rcs, obj, fbx y ortho

//////////////////

curl -v 'https://developer.api.autodesk.com/photo-to-3d/v1/photoscene'

-X 'POST'

-H 'Authorization: Bearer ${toke\_access}'

-d 'scenename=pruebaObj'

-d 'format=obj'

//////////////////

**paso 2 una ves obtenido $photosceneID'**

nota:

* se procese a cargar las img en lotes de 20
* siempre se debe usar el mismo phosheneid
* se pueden cargar enlaces de img (exito) o img guardas localmente (sin éxito )

/////////////////

curl -v 'https://developer.api.autodesk.com/photo-to-3d/v1/file'

-X 'POST'

-H 'Authorization: Bearer ${token\_access}'

-F "photosceneid=${photosceneID}"

-F "type=image"

-F "file[0]=https://raw.githubusercontent.com/FreitesGuillermo/ive/DSC\_5427.JPG"

-F "file[1]=https://raw.githubusercontent.com/FreitesGuillermo/ive/DSC\_5428.JPG"

////////////////

**paso 3 procesar foto scena**

////////////////

curl -v 'https://developer.api.autodesk.com/photo-to-3d/v1/photoscene/photosceneID'

-X 'POST'

-H 'Authorization: Bearer ${token\_access}'

/////////////////

**paso 4 esperar a que se prosece (100%)**

/////////////////

curl -v 'https://developer.api.autodesk.com/photo-to-3d/v1/photoscene/photosceneID/progress'

-X 'GET'

-H 'Authorization: Bearer ${token\_access}'

////////////////

**paso 5 solicitar las fotos procesadas**

nota solo se puede solicitar el tipo de dato que se especificó en el paso 1

se obtiene un enlace que dura 7 días en Amazon y una vez pasado ese tiempo se eliminan

////////////////

curl -v 'https://developer.api.autodesk.com/photo-to-3d/v1/photoscene/photosceneID?format=obj'

-X 'GET'

-H 'Authorization: Bearer ${access\_token}'

**Capítulo 4**

Guardar modelos en cache para usar sin conexión a internet

Una vez aplicación esté finalizada se puede añadir la herramienta de PWA

Progresive web aplicacition: permite otorgar ciertas caracterices móviles a sitios web

Como:

* instalarse en equipos móviles y de escritorio
* operar sin conexión

en este caso se ejecuta el comando:

**ng add @angular/pwa**

donde se crean todos los documentos necesarios,

ejecutar build de producción

para cachar los modelos de VIewer se debe modificar el documento ngsw-config.json

 "dataGroups": [

    {

    "name": "code",

    "urls": ["/api/\*\*","./\*\*","/\*\*"],

    "cacheConfig": {

      "strategy": "freshness",

      "maxAge": "3d",

      "maxSize": 100,

      "timeout": "10s"

    }

  }

  ]

Limitantes o problemas para tener en cuenta:

* solo se puede ejecutar en https
* se tiene que refrescar en varias ocasiones página para que se puedan reflejar los cambios o vaciar la cache
* para realizar pruebas de forma local se debe de ejecutar en producción y utilizar la herramienta de “ npm i -g angular-http-server ” luego en la carpeta /dist/[nombredelProyecto] se ejecuta el comando angular-http-serve