JavaScript 2021

Práctica 1

Objetivos

- Establecer el entorno de trabajo que se utilizará a lo largo de la materia.
- Entender los principios básicos de git.
- Conocer los entornos de desarrollo del navegador y sus herramientas de depuración.
- Realizar operaciones básicas con JS.

Ejercicio 1

Instale y configure las herramientas necesarias para la resolución de los ejercicios de la práctica. Elija uno de los métodos propuestos y siga los pasos indicados.

Herramientas que vamos a utilizar

1. Un Navegador: Chrome, Chromium, Firefox, Opera, etc.
2. npm: gestor de paquetes para node.
3. git: un gestor de control de versiones. Guía.

Método 1

La cátedra provee una máquina virtual que integra todas las herramientas y tiene los archivos base para comenzar.

1. Descargue la máquina virtual de la página de Cátedras.
2. Descargue e instale VirtualBox y **Oracle VM VirtualBox Extension Pack**.
3. Desde el menú "Archivo" de VirtualBox, elija la opción "Importar servicio virtualizado..." para importar la máquina virtual.
4. Verifique en las propiedades de la máquina virtual, las opciones de RAM entre otras.
5. Arranque la máquina virtual y use la contraseña "js21" para ingresar.
6. Abra una terminal y arranque el servidor local que está en la carpeta js21-static con el comando "npm start"
7. Dentro de la máquina virtual navegue con Firefox o Chromium a la página de ejemplo: http://localhost:3000/.
8. Modifique el archivo de ejemplo y recargue la página desde el navegador.
9. En la terminal, detenga el servidor con la combinación de teclas CTRL+C.

Método 2

Con Docker (necesita la carpeta js21-static-docker e instalar git).

Instalación en Windows:

1. Requerimientos: Microsoft Windows 10 Professional o Enterprise 64-bit, O Windows 10 Home 64-bit con WSL 2 (pasos método 2).

2. Descargue desde aquí
3. Para más información recordá visitar y seguir la guía oficial
4. Abra la terminal (PowerShell), diríjase al directorio *js21-static-docker* y ejecute el comando inicial:

```
docker-compose run backend npm install
```

Instalación en macOS:

1. Requerimientos: macOS debe ser la versión 10.14 o más reciente
2. Descargue desde aquí
3. Para más información recordá visitar y seguir la guía oficial
4. Abra la terminal, diríjase al directorio *js21-static-docker* y ejecute el comando inicial:

```
docker-compose run backend npm install
```

Instalación GNU/Linux:

1. Requerimientos: Ampliamente Soportado
2. La instalación depende de la versión. Se puede encontrar la guía de cada una en sublink "Installation per distro"
3. Acceda a la documentación aquí
4. Es importante que lea y asigne permisos al usuario como se describe en la guía de post-instalación
5. También con esta instalación es necesario que instale el compose de docker. Lo puede encontrar en subpestaña Linux
6. Abra la terminal, diríjase al directorio *js21-static-docker* y ejecute el comando inicial:

```
docker-compose run backend npm install
```

Ejecución en todas las plataformas

1. Abra una terminal, diríjase al directorio *js21-static-docker* y ejecute:

```
docker-compose up
```

2. Abra su navegador e ingrese la dirección de la página de ejemplo: http://localhost:3000/.
3. Modifique el archivo de ejemplo y recargue la página desde el navegador.
4. En la terminal, detenga el servidor con la combinación de teclas CTRL+C.

Método 3

Con el Subsistema de Windows para Linux (WSL).

1. En **Windows 10** active la característica de Windows *Subsistema de Windows para Linux*.

2. En el Microsoft Store busque e instale la aplicación Ubuntu 20.04. Es una descarga de 400 megas.
3. Ejecute la aplicación Ubuntu y proporcione un usuario y contraseña.
4. Desde el Microsoft Store instale la aplicación Windows Terminal.
5. Ejecute Windows Terminal y abra un nuevo tab con una consola de Ubuntu desde el menú.
6. Instale nvm ejecutando:

```
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.37.2/install.sh |
bash
```

7. Cierre y vuelva a abrir la terminal
8. Verifique que nvm esté correctamente instalado ejecutando el comando

```
nvm -v
```

9. Instale node

```
nvm install node
```

10. Verifique que se haya instalado node correctamente con el comando:

```
nvm ls
```

11. Descargue el código de ejemplo **js21-static** disponible en Cátedras
12. Descomprima el contenido en una carpeta y ejecute

```
npm install
```

13. Arranque el servidor con el comando "npm start"
14. Navegue a la página de ejemplo: http://localhost:3000/.
15. Modifique el archivo de ejemplo y recargue la página desde el navegador.
16. En la terminal, detenga el servidor con la combinación de teclas CTRL+C.

- Más detalles acerca de cómo habilitar Windows Subsystem for Linux.
- Más detalles acerca de cómo instalar nvm, node, etc. luego de instalar WSL.

Método 4

En Ubuntu (similar para otras distribuciones de Linux)

1. Instale git y curl

```
sudo apt-get install git curl
```

2. Instale nvm:

```
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.37.2/install.sh |
bash
```

3. Cierre y vuelva a abrir la terminal (o cierre la sesión y vuelva a ingresar)
4. Verifique que nvm esté correctamente instalado ejecutando el comando

```
nvm -v
```

5. Instale node

```
nvm install node
```

6. Verifique que se haya instalado node correctamente con el comando:

```
nvm ls
```

7. Descargue el código de ejemplo **js21-static** disponible en Cátedras
8. Descomprima el contenido en una carpeta y ejecute

```
npm install
```

9. Arranque el servidor

```
npm start
```

10. Navegue a la página de ejemplo: http://localhost:3000/.
11. Modifique el archivo de ejemplo y recargue la página desde el navegador.
12. En la terminal, detenga el servidor con la combinación de teclas CTRL+C.

Ejercicio 2

Comenzando con HTML.

1. Dentro de la carpeta **public** (que está en js21-static) cree un archivo llamado {nombreAlumno}-dashboard.html.
2. En ese documento HTML vaya incorporando (por ejemplo, mediante una lista) los vínculos a los demás ejercicios de la práctica a medida que los resuelva. Puede tener un documento HTML por cada ejercicio o varios ejercicios en un mismo documento.
3. Cree en la carpeta **public** una subcarpeta llamada **css** y dentro de ella un archivo llamado {nombreAlumno}-dashboard.css y enlace este archivo al documento HTML.

Ejercicio 3

Hola mundo! utilizando

```
    console.log()
```

1. Dentro del archivo dashboard.html agregue un elemento script que use la función console.log para imprimir el mensaje "Hola Mundo!".
2. Accediendo al documento mediante la URL local (http://localhost:3000) use las herramientas de desarrollo web de Firefox y Chromium para verificar que se haya ejecutado el script (pestaña 'console').

Ejercicio 4

Debug

1. Modifique el script del ejercicio 3 agregándole la siguiente sentencia.

```
debugger;
```

2. Accediendo al documento mediante la URL local (http://localhost:3000) use las herramientas de desarrollo web de Firefox y Chromium para ver qué efecto tiene el cambio realizado.

Ejercicio 5

Link

1. Cree una carpeta llamada js dentro de js21-static/public
2. Cree un archivo llamado practica1.js y guárdelo en la carpeta js21-static/public/js/
3. Dentro de ese archivo defina una función como la siguiente.

```
    function log(message) {
        console.log(message);
    }
```

4. Modifique el dashboard para vincular el archivo practica1.js.

5. Utilice la función log con el mismo mensaje del ejercicio 3 desde el documento html y compruebe el resultado en las herramientas de desarrollo web.

Ejercicio 6

Trabajando con strings (Variables de Texto)

1. Dentro del script creado en el Ejercicio 5 en dashboard.html, cree una variable text y como valor pegue un texto Lorem Ipsum de 5 palabras (puede generarlo desde aquí)
2. Dentro de practica1.js cree una función llamada ejercicio6 que reciba como parámetro una variable e informe por consola utilizando la función del ejercicio 5 la cantidad de caracteres que contiene la cadena.
3. Agregue a la función creada anteriormente otro informe en consola con la posición en que comienza la palabra "ipsum" si es que existe.
4. Asigne en una variable local de la función el substring desde la posición 1 a la 4 e imprima en consola el resultado en mayúsculas.

Ejercicio 7

Trabajando con Números (Variables Numéricas, Operaciones Matemáticas)

1. Dentro de practica1.js cree 3 constantes (A, B y C) y a cada una asigne un valor numérico.
2. Cree una función ejercicio7 en la cual realice e informe por consola la operación A + B elevado a la C (usa la suma de A y B como base y C como exponente).
3. Modifique la asignación de los números (hardcodeados) por una asignación más dinámica de valores enteros utilizando la función random y floor de Math (en las 3 variables).
4. Agregue a la función ejercicio7 la impresión en consola del número más grande de las 3 variables.

Ejercicio 8

Trabajando con Fechas (consulta de documentación en Fechas Nativas)

1. Cree 2 variables (dia1 y dia2). A dia1 créelo con la fecha actual y a dia2 con el valor 1575978300000 (en timestamp).
2. Cree una función imprimirFecha en la que informe por consola el día recibido como parámetro en formato dd/mm/aaaa hh:ss. Llame a la función declarada con el valor dia2 como parámetro.
3. Cree una función donde reciba 2 parámetros de fecha e internamente asigne al dia2 el año del dia1 y en dia1 establezca como mes el del dia2. Imprima los valores obtenidos utilizando la función imprimirFecha
4. Realice una función que reciba dos fechas como parámetro y las reste retornando una nueva fecha con la diferencia entre ambas.
5. Llame a la función antes creada e imprima en consola el resultado de la misma en días.

Ejercicio 9

Gitlab

1. En su máquina o en la máquina virtual (según como hayan resuelto el punto 1) genere la clave ssh para poder acceder desde esa máquina al repositorio git.

```
ssh-keygen -t rsa -b 2048 -C "tu@email.com"
```

2. Imprima en pantalla la clave generada. **Nota: "js21" es el nombre de usuario de la máquina en la que está trabajando.**

```
cat /home/{nombreDeUsuario}/.ssh/id_rsa.pub
```

3. Copie la clave (es un único renglón) y agréguela en Gitlab dentro de Settings / SSH Keys
4. Modifique la configuración global de git con sus datos

```
git config --global user.name "Nombre y Apellido"
git config --global user.email "tu@email.com"
```

5. Puede también configurar su editor favorito para realizar los commit (opcional)

```
git config --global core.editor={vim, nano, vscode, vi, gedit, etc}
```

Puede ver qué otras configuraciones se pueden establecer usando:

```
git config --list
```

6. clone el repositorio de su grupo (reemplace # por su número de grupo en {grupo-#})

```
git clone git@gitlab.catedras.linti.unlp.edu.ar:js/estudiantes/{grupo-
#}.git
```

Ejercicio 10

Git colaborativo (se recomienda leer un poco más de git aquí)

1. Verifique el estado del área de trabajo local con el comando

```
git status
```

2. Si el comando devuelve "Tu rama está actualizada con 'origin/master'." continúe con el punto 3, si no el comando debería devolver "Tu rama está detrás de 'origin/master' ..." para esto ejecute el siguiente comando que lo que hace es traer los cambios que están en el repositorio remoto. El 2do del grupo que hace el commit seguro tiene que hacer este paso).

```
git pull
```

3. Agregue lo archivos que desea subir al repositorio: ("." agrega todos, si no pueden agregar de a uno con el path relativo)

```
git add .
```

4. Haga el *commit* de los cambios en el repositorio local proporcionando un mensaje:

```
git commit -m "primer commit, {nombre del alumno}"
```

si se equivocó en el mensaje y quiere arreglarlo use

```
git commit --amend
```

5. Suba los cambios del repositorio local al repositorio remoto:

```
git push -u origin master
```

6. En cualquier momento puede ver el historial del repositorio haciendo:

```
git log
```

Ejercicio 11

Subiendo cambios en una rama de Git

Primero lo que debe hacer es crear una rama propia con el nombre completo de cada uno con la forma: nombre-apellido. Para esto ejecute

```
git checkout -b nombre-apellido
```

Esto crea una rama y lo posiciona sobre ella. Como todos los cambios realizados hasta aquí no tienen confirmación (git commit) estos cambios los lleva directamente a la rama. Para corroborar que esto es así y que está posicionado sobre la rama correcta ejecute

```
git status
```

Esto informa en la primera línea en qué rama está posicionado y casi a lo último los cambios que ha realizado en los archivos (color rojo). Agregue los archivos para subir

```
git add .
```

Si hace git status puede ver los cambios en verde. Realice el *commit*

```
git commit -m 'subiendo la práctica 1 a mi rama'
```

Como la rama no está creada en el repositorio remoto, tiene que agregar por única vez el parámetro *--set-upstream*

```
git push --set-upstream origin nombre-apellido
```