Code Docs

# General

## Temp

## Run Node app on Windows startup

* Use npm and install node-windows
* Create and run this script

var Service = require('node-windows').Service;

// Create a new service object

var svc = new Service({

  name:'Hello World',

  description: 'The nodejs.org example web server.',

  script: 'C:\\path\\to\\helloworld.js'

});

// Listen for the "install" event, which indicates the

// process is available as a service.

svc.on('install',function(){

  svc.start();

});

svc.install();

* A service will be found in windows services that can be made to run automatically on startup

## Certificates

* A .pfx file includes both public and private keys for the associated certificate. Never share this file. It can be used for signing and authentication.
* A .cer file only has the public key. It can be used for authentication.
* Trusting a setup file:
  + Install the certificate used to sign the file into Trusted People in Local Machine

### Creating a self-signed certificate

* Open powershell

$cert = New-SelfSignedCertificate -DnsName <publisher> -Type CodeSigning -CertStoreLocation cert:\CurrentUser\My -NotAfter (Get-Date).AddYears(10)

* The –DnsName becomes certificate common name (CN). It has to match the publisher in electron builder for the signing to work.
* Exporting the certificate
  + Create a password first. Exporting is not allowed before this

$CertPassword = ConvertTo-SecureString -String "<password>" -Force –AsPlainText

* Export

Export-PfxCertificate -Cert "cert:\CurrentUser\My\$($cert.Thumbprint)" -FilePath "<path/to/selfsigncert.pfx>" -Password $CertPassword

### View existing certificates

* Open powershell

cd cert:\CurrentUser\My

# Language Guides

## Js

### General

* == checks if values are equal. === checks if values are equal and of same type.

### Printing output

* Template strings

console.log(`story ${variableName} story`);

* Format print

console.log('story %s story',name);

## SVG

* The viewBox attribute can be used to scale SVG elements

### SVG Scrolling

* If SVG has elements in it that are bigger than the SVG size, they will get clipped by default.
  + They can be made to be shown by setting the overflow property to visible.
    - Setting the property to scroll does not create scrollbars and doesn’t seem to work.
* If the SVG is bigger than it’s container div, the div can be set to allow scroll of overflow.

# Library Guides

## D3 Js

### Sending event from one element to another

For example, when you want to send a mousewheel event from element 1 to element 2. You can clone the event (you can’t reuse the already fired event) and use the dispatchEvent function.

The dispatchEvent function is only available for some types of elements like svg rect.

element1.on("wheel",(e)=>{element2.node().dispatchEvent(new WheelEvent(e.type,e))})

### Getting parent node of an element

node.parentNode

### Hide elements and prevent events

elem.style("display","none")

To undo

elem.style("display",null)

# Framework Guides

## Node

### Update modules to latest

* Install npm-check-updates

npm install –g npm-check-updates

* Run it

ncu –u

* This will update package.json with the latest versions
* Now run update

npm update

## Check if module is being run as a script

if (typeof require !== 'undefined' && require.main === module) {

// Script code

}

# Tool Guides

## Git

### Cloning your own repo on github

git clone https://github.com/userName/Repo New\_Repo

cd New\_Repo

git remote set-url origin https://github.com/userName/New\_Repo

git remote add upstream https://github.com/userName/Repo

git push origin master

### Moving a submodule

git mv old/path new/path