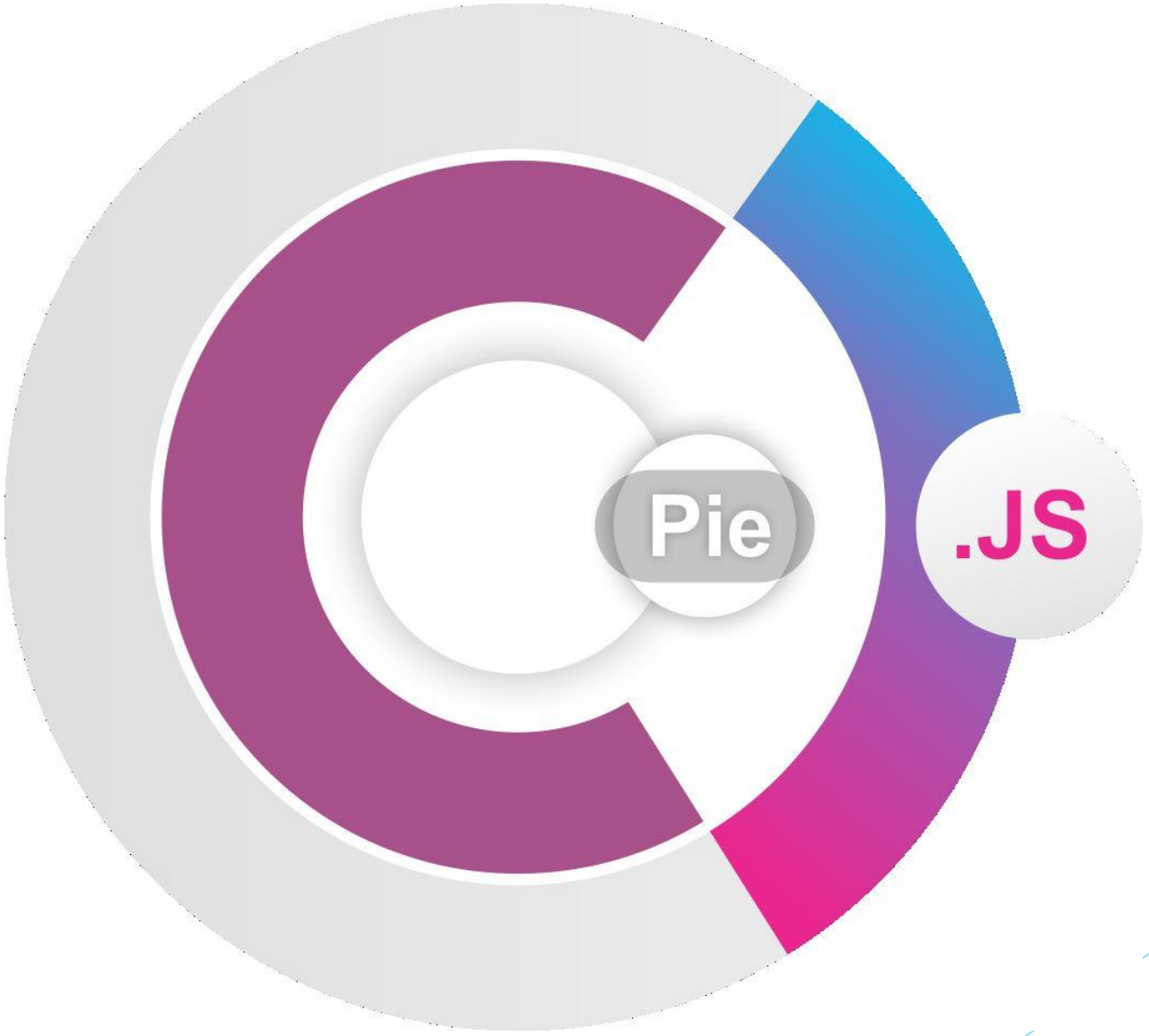

Pie
.JS

THE PIE FRAMEWORK

PieJS 1.0+ Beta
Documentation by Eugy Enoch

INTRO

PieJS(Pie-JS) or pie.js is a modern and SEO friendly JavaScript library and part of the Pie Framework which also includes The PieCSS or pie.css technology. It is still being developed and it is intended to be among the most used JS frameworks for fast development of websites and web applications that are targeted for all devices and suitable for any and every occasions. PieJS is also intended to add speed and make learning and using the framework and technology easy for all.

AVAILABILITY AND COST

Availability of the PieJS framework is intended for free and at no costs at all. The PieJS framework can be downloaded from the internet at authorized points and used freely for building websites and web applications.
Begin here => https://eugyenoch.github.io/pie/

LICENSING

The PieJS library is licensed under the MIT license.
Read more about the MIT license here:
https://github.com/eugyenoch/pie/blob/master/LICENSE

THE DEVELOPER

The initial developer for pie is an experienced, talented and exceptional leader committed to maintaining cutting edge technical and relationship skills and up-to-date industry knowledge; with outstanding relationship building, training and presentation skills and excellent at juggling multiple tasks and working under pressure. Broad industry experience includes Education, ICT, Books Publishing/Promotion, Media, Sales, Administration, Charity and Religion.
David Ugochukwu Enoch(Eugy Enoch) currently works at the APTECH Education Center as a Faculty and welcomes collaborators and issues rising out of the usage of the PieJS software.
Reach him on Github + Twitter + LinkedIn + Facebook + Telegram + Instagram = @eugyenoch

COLLABORATORS AND SPONSORS

Sponsors are welcome to contribute via any number of ways and Collaborators are welcome to contribute via Github to the PieJS library in any ways possible including but not limited to new issues, commits and other contributions. Collaborators and sponsors are treated as co-owners in the project.

ISSUE REPORTS

Reporting issues for now can be done through the Pie Github platform and on all Ojaay platforms.

JQUERY AS USED IN THE PIEJS FRAMEWORK

The essence of the JQuery component in the PieJS framework is to jQuery is to make things like HTML document manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers including with it a combination of versatility and extensibility.

The JQuery Framework/Library is authored originally by John Resig and released under the MIT license.

WHY USE PIEJS

Advantages of PieJS:

Although being worked, using pie.js comes with some benefits that is better explored by the user:

- PieJS is built to be and is {absolutely} SEO, SMO and digital marketing friendly
- Easy to use by anybody with even the faintest knowledge of HTML, CSS and JavaScript
- PieJS is compatible with all modern browsers be it Firefox, Edge, Chrome, Opera, UCBrowser, Safari or the myriads of desktop and mobile browsers that implement modern standards and API
- The pie frameworks, generally, allow users build prototypes of a solution quickly.

INSTALLATION

General

PieJS can be downloaded from the website. The files available for download are:

- Uncompressed – Intended for development
- Compressed – Intended for production

The pie.js library can also be downloaded on github

On NodeJS

The PieJS library can also be used on NodeJS in which case NodeJS will have to be installed before using PieJS.
There is a guide on the internet on installing node.js on your machine
https://phoenixnap.com/kb/install-node-js-npm-on-windows

THE PEOPLE BEHIND PIEJS

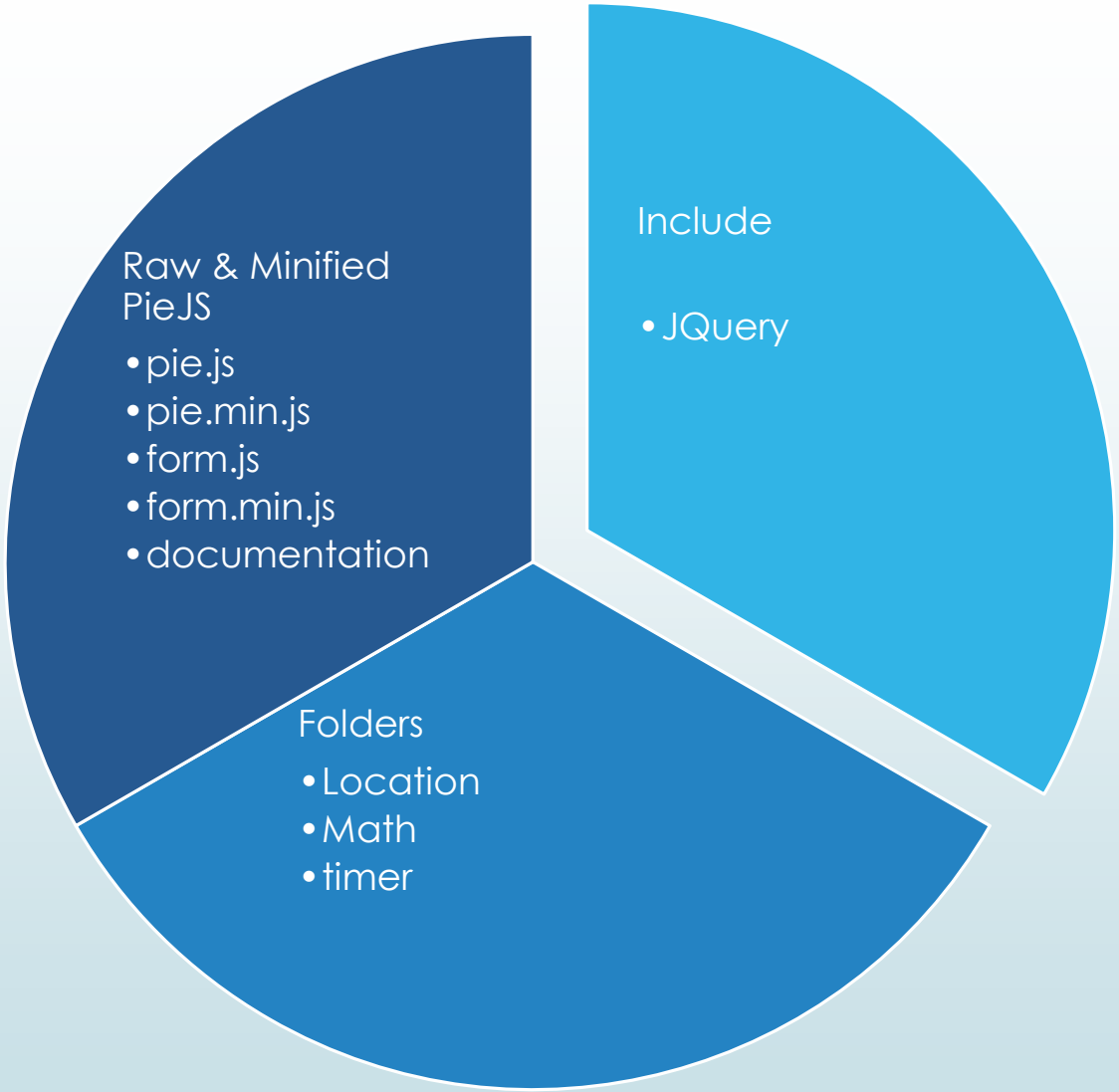
The team at the Pie Framework
Github => https://eugyenoch.github.io/pie/
Facebook => https://www.facebook.com/ojaay
Twitter => https://twitter.com/ojaay
Linkedin => https://www.linkedin.com/company/ojaay

Founder
Github: https://github.com/eugyenoch
Twitter: https://twitter.com/eugyenoch
Facebook: https://www.facebook.com/eugyenoch
Linkedin: https://www.linkedin.com/in/eugyenoch

An aside => Are you a JavaScript guru or enthusiast? We want you to join us, to contribute to the continued development of the PieJS framework. We appreciate your coming a lot and look forward to working with you.

STRUCTURE

The preambles and more



The PieJS Folder Structure for v1.0+ Beta Release

CONTENT OF THE PIEJS FRAMEWORK

PieJS, just like PieCSS, adopts a simple folder structure and an easy to understand file naming pattern.

FOLDER STRUCTURE AND CONTENT

The following folders are available:

- **Piejs** => root folder for the piejs framework and also contains the files: *pie.js; pie.min.js; preloader.js* and some licensing.
- **Include** => contains dependencies that are needed for parts of the pie.js functionality. Inside this folder are files like *JQuery(version 3.4.1 shipped along)*
- **Doc** => Contains documentation for the PieJS(pie.js) framework. Please note that the documentation, though , a very good guide is not intended to replace learning and certification in the pie framework.

Watch out for official trainings and certification in the Pie frameworks, meanwhile send any enquiries to the following contact information and more as may be revealed officially in future dates by the Pie team. You should also follow these social media handles.

An aside => All non-minified files that come with the pie framework are commented enough and are intended to guide you even further in appreciating the work done in and using the Pie framework.

Please always use `<meta name="viewport" content="width=device-width,initial-scale=1.0" />` in your `<head>` sections to sue for responsiveness and standard.

GETTING STARTED

Guide to getting started

GUIDE TO GETTING STARTED

The PieJS folders contain the files needed for your website and anyone may link to one or more of the files present as need demands for their projects.

HOW TO

Using downloaded pie.js

Just like regular JavaScript, PieJS and it's associated files can be linked to a HTML document appropriately, using the `<script> </script>` tags placed within the `<head>` and/or `<body>` section of a `<html>` document. The syntax is shown below:

```
<script src="piejs/pie.js" type="text/JavaScript"> </script>
```

The "`type=text/JavaScript`" is, nowadays, optional and hence not required at all as JavaScript is now the default styling language of the web and for most browsers as well so the above linking can be refined and presented preferably thus:

```
<script src="piejs/pie.js"> </script>
```

Linking up online pie.js

Just like regular JavaScript, PieJS and it's associated files can also be linked to a HTML document online using the <script> </script> tags as well placed within the <head> and/or <body> section of a <html> document. The syntax is shown below:

```
<script src="pie.js" type="text/JavaScript"> </script>
```

The "type=text/JavaScript" is, nowadays, optional and hence not required at all as JavaScript is now the default styling language of the web and for most browsers as well so the above linking can be refined and presented preferably thus:

```
<script src="pie.js"> </script>
```

WORKING WITH ELEMENTS

Understanding the PieJS Elements Presentation system
For HTML and CSS elements

While the pieJS continue to be developed, there are currently the following possible on piejs alone.

❑ SELECTING ELEMENTS AND PERFORMING ACTIONS ON THEM

Selecting elements

The `pie.action(selector)` selects elements and performs actions on them. For example
Elements selected are placed in parenthesis and is compatible with conventional JavaScript element selection rules. For example:

(“*”) => Selects all elements in the document.

`(this)` => Selects the current element

(“p”)=> Selects all <p> elements

(“span.new”)=> Selects all elements with class named ‘new’

(“div span”)=> Selects all elements inside all elements

(“img[title=]”)=> Selects all <img> elements with `title` unspecified

(“[title]”)=> Selects all elements with `title` attributes

Performing actions on elements

Performing actions on selected elements in piejs is compatible with conventional JavaScript element selection rules. For example:

`pie.hide('h1')` => This will hide all `<h1>` elements.
`pie.show('h1')` => This will show all `<h1>` elements.

Element identifiers can also be selected and have an action performed on them in piejs. For example

`pie.hide('.myPie')` => This will hide all elements with the class 'myPie'
`pie.hide('#myPie')` => This will hide all elements with the id 'myPie'
`pie.show('.myPie')` => This will show all elements with the class 'myPie'
`pie.show('#myPie')` => This will show all elements with the id 'myPie'

❑ SLIDERS

A slideshow can be done with piejs as using the `.img-slider` class and calling the `pie.imgslider( )` function on an element.

To achieve this, simply add the `.img-slider` class to all images that are to slide and PieJS does the rest

❑ LIST TOGGLER

A list toggle will determine how your lists are going to slide out to view to users. PieJS supports two(2) lists toggle method like so:

1. **The tree arrow toggle method**
   To use this method, simply open a `<ul>` or `<ol>` element and use the class `.tree-view-arrow` in a *main* `<li>` item underneath (preferably the first and some other list items you want to nest other list items underneath). Thereafter open another `<ul>` or `<ol>` element with class `.nested` and add as many `<li> list items</li>` as required. Thereafter close all other open `<ul>` or `<ol>` and main `<li>` items to achieve the desired effects.
2. **The tree check toggle method**
   To use this method, simply open a `<ul>` or `<ol>` element and use the class `.tree-view-check` in a *main* `<li>` item underneath (preferably the first and some other list items you want to nest other list items underneath). Thereafter open another `<ul>` or `<ol>` element with class `.nested` and add as many `<li> list items</li>` as required. Thereafter close all other open `<ul>` or `<ol>` and main `<li>` items to achieve the desired effects.

❑ FORMS

Forms is an element that is capable of accepting inputs from users, and, buttons are used to effect an action, on a form or otherwise outside of a form.

Popout forms

Call the `pie.openForm()` and `pie.openForm()` functions to open and close a popout form respectively.

Both should be used via an element with an `id="popForm"`, that is, on `#popForm`

❑ Form Manipulation

Manipulating forms using pieJS is possible as so:

Form function	Description	Id/Class where applicable
`frm.reset()`	Resets form data	Form must have id="freset"
`frm.submit()`	Submits form data	Form must have id="fsubmit"
`frm.elvalue()`	Display the value of each form element	Form must have is="fevalue" and another element where form entries will be displayed have an id="displayFrom"
`frm.disableView()`	Disable options list	<select> and <option> list must have a id="formSelect"
`frm.ableView()`	Enable options list	<select> and <option> list must have a id="formSelect"
`frm.multipleSelect()`	Enable multiple select	<select> and <option> list must have a id="formSelect"

❑ FORMS Cont'd

Form function	Description	Id/Class where applicable
`frm.sizeTwo()`	Convert to multiselect mode - change size to 2	<select> and <option> list must have a id="formSelect" and must have up to 2 options
`frm.sizeFour()`	Convert to multiselect mode - change size to 4	<select> and <option> list must have a id="formSelect" and must have up to 4 options
`frm.sizeEight()`	Convert to multiselect mode - change size to 8	<select> and <option> list must have a id="formSelect" and must have up to 8 options
`frm.sizeTen()`	Convert to multiselect mode - change size to 10	<select> and <option> list must have a id="formSelect" and must have up to 10 options
`frm.sizeTwelve()`	Convert to multiselect mode - change size to 12	<select> and <option> list must have a id="formSelect" and must have up to 12 options
`frm.sizeSixteen()`	Convert to multiselect mode - change size to 16	<select> and <option> list must have a id="formSelect" and must have up to 16 options

❑ FORMS Cont'd

Form function	Description	Id/Class where applicable
`frm.optRemove()`	Removes selected options	<select> and <option> list must have a id="formSelect" and must have up to 10 options
`frm.optIndex()`	Display index of selected option	<select> and <option> list must have a id="formSelect" and must have up to 16 options and another element where form entries will be displayed have an id="dispOptIndex"
`frm.outOption()`	Output all selected options	<select> and <option> list must have a id="formSelect" and must have up to 16 options and another element where form entries will be displayed have an id="dispOutOption"
`frm.dispOption()`	Display selected options	<select> and <option> list must have a id="formSelect" and another element where form entries will be displayed have an id="dispFormSelect"

WORKING WITH LOCATION

Using Geolocation, History objects, Reloads and Redirects(HREFs, Assign, Replace functions) in PieJS

❑ GEOLOCATION

In web development, The Geolocation object allows the user to provide their location to web applications. In most cases and for privacy concerns, the user is asked for permission to report location information. Geolocation is most accurate for devices with GPS and is mostly served over the https: protocol.

In JavaScript, there is a reload() method that does the same as the reload button in your browser, however, there may be times when a page is preferred to reload from server and in this case, call the `reloadServerDoc()` function on an event. If however, a cache based page reload is needed, then the `reloadCacheDoc()` function should be called instead.

To use geolocation in PieJS, make sure the appropriate script – `geolocation.js` – is linked in the current web page, as so.:

```
<script src="piejs/location/geolocation.js"></script>
```

Note: If the geolocation.js file was moved to a different folder, it is advised to indicate the correct file path.

❑ HISTORY OBJECTS

The history object contains URLs visited by a user a user can choose to go back or forward to previously loaded elements or URLs Important note.

The history functions can be useful in building applications that need to move back or forward in calculated single or double steps.

GOING BACK

To go back one step in memory history; call the `goBack()` function on an element.
To go back two steps; call the `goBackDouble()` function.

GOING FORWARD

To go forward one step in memory history; call the `goForward()` function on an element.
To go forward two steps; call the `goForwardDouble()` function.

The file to communicate with is the `history.js` file and is linked normally as so:

```
<script src="piejs/location/history.js"></script>
```

NOTE: If the history.js file was moved to a different folder, it is advised to indicate the correct file path.

❑ RELOAD

In web development, a reload refreshes the current web page in view to serve another version of the web page in view from the web server or cache.

In JavaScript, there is a reload() method that does the same as the reload button in your browser, however, there may be times when a page is preferred to reload from server and in this case, call the `reloadServerDoc()` function on an event. If however, a cache based page reload is needed, then the `reloadCacheDoc()` function should be called instead.

To use reload in PieJS, make sure the appropriate script – `reload.js` – is linked in the current web page, as so: `<script src="piejs/location/reload.js"></script>`

Note: If the reload.js file was moved to a different folder, it is advised to indicate the correct file path.

❑ REDIRECT

A redirect does what it says; moves the user from the current web page in view to a new called web page.

In PieJS, there are three(3) coded methods to force a page or section redirect.

First method is the invocation of the `locationHref()` function on an event which stimulates a mouse click on an object/element. If you are redirecting after an AJAX call completes, keeping the originating page in history might be expected, in which case, the `locationHref()` function would be a recommended option.

Second method is the invocation of the `locationReplace()` function on an event which stimulates an HTTP redirect.

This method also removes the URL of the current document from the document history meaning that it is not possible to use the "back" button to navigate back to the original document. If this is your intention, then the `locationReplace()` function would be a recommended option.

❑ REDIRECT CONT'D

Third method is the invocation of the `locationAssign()` function on an event which also stimulates an HTTP redirect.
This method, unlike the second method, does not removes the URL of the current document from the document history meaning that it is possible to use the "back" button to navigate back to the original document. If this is your intention, then the `locationAssign()` function would be a recommended option.

Important note
To use any of the redirects, open up the appropriate js file and edit the custom url you wish to redirect to. Replace the word "YOUR_URL" with the actual URL you would love to redirect users to, leaving the double quotes " " in place.

To use any of redirects in PieJS, make sure the appropriate script – `redirect.js` – is linked in the current web page, as so: `<script src="piejs/location/redirect.js"></script>`

If the redirect.js file was moved to a different folder, it is advised to indicate the correct file path.

WORKING WITH MATH

Using Math functions in PieJS

❑ JS Math

Mathematical functions have been built in to the piejs framework.

JS Calculator as used in PieJS

The Javascript developed by Eugy Enoch originally for use in the Eugy calculator software program and now included in the PieJS build.

Math Functions

A few functions(to embed with web elements and events) to get you going are listed as follows:

Math Function	Description
`calc.rad()`	Displays the radian of the calculated value
`calc.rand()`	Generated random number(preceded by text)
`calc.increase()`	Increments displayed number by 1
`calc.decrease()`	Increments displayed number by 1
`calc.log2()`	Calculate the natural log2 of the value displayed
`calc.log10()`	Calculate the natural log10 of the value displayed
`calc.abs()`	Calculates the absolute of the value in display

Math Functions cont'd

Math Function	Description
`calc.raise2()`	Calculates the display to the second power (raise to power 2)
`calc.raise3()`	Calculates the display to the third power (raise to power 3)
`calc.max()`	Returns the maximum from a range
`calc.min()`	Returns the minimum from a range
`calc.round()`	Rounds up the displayed value
`calc.pi()`	Displays the value of pi
`calc.sin()`	Calculates the sine(in radian degrees) of the displayed value
`calc.asin()`	Calculates the arcsine(in radian degrees) of the displayed value
`calc.cos()`	Calculates the cosine(in radian degrees) of the displayed value
`calc.acos()`	Calculates the arccosine(in radian degrees) of the displayed value
`calc.tan()`	Calculates the tangent(in radian degrees) of the displayed value
`calc.atan()`	Calculates the arctangent(in radian degrees) of the displayed value

Math Functions cont'd

Math Function	Description
`calc.trunc()`	Rounds the the displayed value
`calc.sqr()`	Returns the square root of the displayed value
`calc.cbrt()`	Returns the cube root of the displayed value
`calc.equal()`	Calculates the displayed expression as a JavaScript expression

NOTE:
To use math functions in PieJS, make sure the appropriate script – `math.pie.js` – is linked in the current web page, as so.:

```
<script src="piejs/math/math.pie.js"></script>
```

Note: If the math.pie.js file was moved to a different folder, it is advised to indicate the correct file path.

WORKING WITH TIMERS

Using Timers and manipulating timing events

❑ Timers

Timers let you keep tab of time and helps control timing events. In Javascript, JS timers can help control timing events. PieJS ships with functions for time and date manipulation methods.

The Timers folder ships with 3 files:

- clockdate.js => creates a digital clock and also function for full date only.
- countdown.js => creates a countdown timer.
- dates.js => creates fulldate and month/year dates.

- **clockdate.js**

Display the clock

This file contains the script to create a digital clock using system time and also include full date in the format(dd/mm/YYYY), when called.

To use; make sure the appropriate script – `clockdate.js` – is linked in the current web page, as so.:

```
<script src="piejs/timer/clockdate.js"></script>
```

Note: If the clockdate.js file was moved to a different folder, it is advised to indicate the correct file path.

After linking with the file, call the `clock.digitClock()` function to include a timer through an event. The timer will appear in an element with `id="digitalClock"`.

Stop clock with clockdate.js

Using the timer from the clockdate.js also has an option to include a stop time to the timer and o achieve this, the `clock.stopClock()` function must be called on any element(HTML button; HTML body; etc) through an event(onload; onclick; etc).

Display the full date only

function for full date is included with the clockdate.js file and can be called using the `clock.dispDate()` function with any event. The full date is displayed on an element with an `id="fullDate"`.

- **countdown.js**

Display the countdown timer
This file contains the script to create a countdown timer to keep track of important events.

To use; make sure the appropriate script – `countdown.js` – is linked in the current web page, as so.:
`<script src="piejs/timer/countdown.js"></script>`

Note: If the countdown.js file was moved to a different folder, it is advised to indicate the correct file path.

After linking with the file. This timer will automatically appear in an element with `id="countdown"`.

By default, this countdown is set to expire on a set date as so: on "Dec 31, 2020 13:27:15" however, this timeframe can be adjusted to any time by simply opening up the countdown.js file and editing the set time frame to any other time and date.

Also by default, the message to display when timer is elapsed is set to "WE ARE HERE NOW". This may also be edited by simply opening up the countdown.js file and editing the message.

countdown timer can be used inside of your web documents to display the countdown timer.
To display a countdown timer, user the Id countdown (#countdown),after linking with the appropriate PieJS document => timer.js as so:

```
<script src="piejs/timer.js"></script>
```

The timer can be used together with CSS to achieve magic and effects.
An example is the timer as used in one of our coming soon page examples as shown below.

```
<div id="countdown" class="padding-xl text-center text-wide font-large text-color-dark-khaki text-color-hover-khaki"></div>
```

- **dates.js**

Display dates
This file contains the script to create display dates only in two formats:
- Full date in the format(dd/mm/YYYY), when called.
- Month and year in the format(mm/YYYY) when called.

To use; make sure the appropriate script – `dates.js` – is linked in the current web page, as so.:
`<script src="piejs/timer/dates.js"></script>`

Note: If dates.js was moved to a different folder, it is advised to indicate the correct file path.

After linking with the file. Call the full date option with the `dates.dispFullDate()` function via any event. The full date is displayed on an element with an `id="fullDates"`.

As an option, call the month/year option using the `dates.monthOnly()` function via any event. The full date is displayed on an element with an `id="fullMonth"`.

Pie is currently maintained by Ojaay and anyone can contribute to PieJS on Github:
https://github.com/eugyenoch/pie