Geogebra Plugin

Geogebra is a graphing and geometry tool I've been using since my teaching days. I always found it to be very useful to display concepts that are hard to visualize on a static page. In the past, I had to export files to html5 to embed them, but the process was pretty clunky and time consuming.

I searched the Dokuwiki plugins but all the ones listed are too old and don't work anymore. Finally, I created my own using some general instructions for embedding Geogebra files. This is my first plugin and a bit of a cheat: I basically took an existing plugin (Color plugin) that was very simple, and modified it until it did what I needed. However, there's still a lot of code that I don't really understand and a few features that are missing.

Here's what I got so far. If anyone has experience writing Dokuwiki plugin, please reach out.

Here's how to installed it:

- Add the following three files in the wiki folder: `./lib/plugins/ggb/`
- `manager.dat`

```
installed=Sun, 08 Nov 2020 08:20:00 -0800
```

- `plugin.info.txt`

```
base   ggb
author Patrick Truchon
email  patoo@rbox.me
date   2020-11-08
name   geogebra6 syntax plugin
desc   Include GeoGebra6 files into Dokuwiki
```

- `syntax.php`

```
<?php
/**
 * Plugin ggb: Embeds Geogebra files into Dokuwiki.
 *
 * @license    GPL 3 (http://www.gnu.org/licenses/gpl.html)
 * @author     Patrick Truchon <patoo@rbox.me>
 * See: https://wiki.geogebra.org/en/Reference:GeoGebra_Apps_Embedding
 */

// must be run within DokuWiki
if(!defined('DOKU_INC')) die();
if(!defined('DOKU_PLUGIN')) define('DOKU_PLUGIN',DOKU_INC.'lib/plugins/');
require_once(DOKU_PLUGIN.'syntax.php');


/**
 * All DokuWiki plugins to extend the parser/rendering mechanism
```

```
 * need to inherit from this class
 */
class syntax_plugin_ggb extends DokuWiki_Syntax_Plugin {

    function getType(){ return 'substition'; }
    function getSort(){ return 306; }
    function connectTo($mode) {
$this->Lexer->addSpecialPattern('{{ggb>.*?}}',$mode,'plugin_ggb'); }


    /**
     * Handle the match
     */
    function handle($match, $state, $pos, Doku_Handler $handler){
        $match = html_entity_decode(substr($match, 6, -2));
        list($ggbfile, $index) = explode('|',$match,2);
        if (preg_match('/(.*) ([0-9]+,[0-9]+)$/',trim($ggbfile),$matches)) {
            $ggbfile = $matches[1];
        if (strpos($matches[2],',') !== false) {
            list($w, $h) = explode(',',$matches[2],2);
            }
        else {
            $w = '800';
            $h = $matches[2];
            }
            }
        else {
            $w = '800';
            $h = '400';
            }

        if (!isset($index)) $index = '';
        return array($ggbfile, hsc(trim($index)), hsc(trim($w)), hsc(trim($h)));
        }

    /**
     * Create output
     */
    function render($mode, Doku_Renderer $renderer, $data) {
        list($ggbfile, $index, $w, $h) = $data;
        if($mode == 'xhtml'){
            $slash = strrpos($ggbfile, "/");
            if($slash > 0){
               $slash = $slash + 1;
               }
            $ggbid = substr($ggbfile, $slash, -4);
            $renderer->doc .= '<script
src="https://www.geogebra.org/apps/deployggb.js"></script>';
            $renderer->doc .= '<div id="'.$ggbid.'"></div>';
            $renderer->doc .= '<script>';
            $renderer->doc .= 'window.addEventListener("load", function() {';
            $renderer->doc .= 'new GGBApplet({"width": '.$w.', "height": '.$h.',
```

```
';
            $renderer->doc .= '"showToolBar": false, "showAlgebraInput": false,';
            $renderer->doc .= '"showMenuBar": false, "filename": "/_media/';
            $renderer->doc .= "$ggbfile\"},true).inject($ggbid) })";
            $renderer->doc .= '</script>';
            $renderer->doc .= 'Download <a
href="/_media/'.$ggbfile.'">'.$ggbid.'.ggb</a>';
            return true;
        }
        return false;
    }
}
?>
```

The syntax is:

```
{{ggb>path/to/geogebrafile.ggb 740,300}}
```

Note that at the moment, if a path is used, it must use / instead of :. The default dimensions are 800×400 if they are omitted.

Here's an example from one of Ham Basics pages showing how different waves add and what SWR looks like:

Wave Addition

When two waves overlap, they add up together at every point. Here, the blue and green waves are generated and add up together to form the red wave. You can move the blue and green waves and see the result. To convince yourself that the red wave is really the sum of the blue and green waves, look at points A, B, and C. You can move the blue or green waves by sliding their phase (φ and Φ) around. You'll see that point C is always the sum of A and B.

Download waveaddition.ggb

Where do the blue and green waves need to be so that...

- the red wave is the biggest?
- the red wave is cancelled out?[1)]

If you press the play button on the bottom left corner, you'll see the blue wave travel to the right and the green wave travel to the left. The red wave, which is the sum of the forward and reflected waves, oscillates up and down but doesn't travel anywhere, which means it's not going into the antenna.

While the animation is running, slowly decrease the amplitude of the reflected wave (V_B) and you'll see that the red wave will start moving to the right. As you do that, notice how the SWR (Standing Wave Ratio) decreases toward 1:1. At this point, there is no reflected wave and all of the energy is going to the antenna (assuming no loss in the feedline).

1)
Fun fact: This is how noise cancelling headphones work. The headset has a microphone that picks up the noise,

inverts the waves, and plays them back in the ear piece. The combination of the real life noise and the inverted noise being played in the speaker cancel out (somewhat).