Pas2JS Integration
Nov. 2023, Max Kleiner

- Combine the world of web development with a desktop development world.
- IDE / RAD / CLI / Shell / Scripts
- Pas2JS and JS1Pas Scripting maXbox4
- https://wiki.freepascal.org/pas2js
- This session shows you various ways of having JS in your application.

EKON 27

Agenda

- JS1Pas – JS integrate in Delphi Form
- Hybrid Mode as TWebBrowser.
- Mapping from JS Lib (ex. unit ChartJS; Pascal mapping for ChartJS: https://www.chartjs.org)
- Pas2Js – from pas source to a *.js
- *Today JS/HTML/CSS/Websockets is a main stack for building frontends and user interfaces. It has great libraries, frameworks and gigantic community.*

EKON 27

From J1Pas to Hybrid

- No Makefiles: Compiler searches for all required source files and RTL's and automatically recompiles all changed files.
- Simple Solution: *OpenWeb(JSApp);*
- Local or Server:
- Const JSAPP2 = 'C:\Program Files\Streaming\IBZ2021\Module2_3\EKON27\text2jstester.html';
- Const JSAPP3 = 'https://raw.githack.com/breitsch2/maXbox4/master/assets/text2jstester.html';

EKON 27

Hybrid Mode

- A hybrid application (hybrid app) is one that combines elements of both native and Web applications.
- In practice it means that you build native applications with a web stack, engine or node. There are several libraries like Electron or NW.js, but we can also build such applications using Delphi or Lazarus.

https://raw.githack.com/breitsch2/maXbox4/master/assets/basicpdf2.html

EKON 27

Hybrid Mode

- ScriptGate for ex. is a cool library that allows to call Delphi methods from JS code in *TWebBrowser*. Also the other way round.
- Now I will create a script application and place webbrowser into a form. Then we add some lines of code:

```
WebBrowser1:= TWebBrowser.create(form1);
  with WebBrowser1 do begin
    TWinControl(WebBrowser1).Name:= 'MyWebBrowser2JS';
    // Parent property is read-only unless cast
    TWinControl(WebBrowser1).Parent:= form1;
```

EKON 27

https://raw.githack.com/breitsch2/maXbox4/master/assets/text2jstester.html

Be aware of

- cFeatureBrowserEmulation =
- 'Microsoft\Internet Explorer\Main\FeatureControl\FEATURE_BROWSER_EMULATION\';
- Late binding possible:
- objIE:= CreateOleObject('InternetExplorer.Application');
- Silent mode to debug and events:

  Silent := False;
- OnDocumentComplete:= @WebBrowserDocumentComplete;

Demo: 1235_Weatherboxsep2023_EKON27_API_JS_Integrate1.txt

API Keys

- Where you store Developer-Keys.
- Use Read-only keys
- These API keys are specifically designed to be used in client-side code. They can only read data from the API, not write to it or change anything. So even someone got a hold of a read-only API key, they couldn't do any damage to your data.

API Key Solution

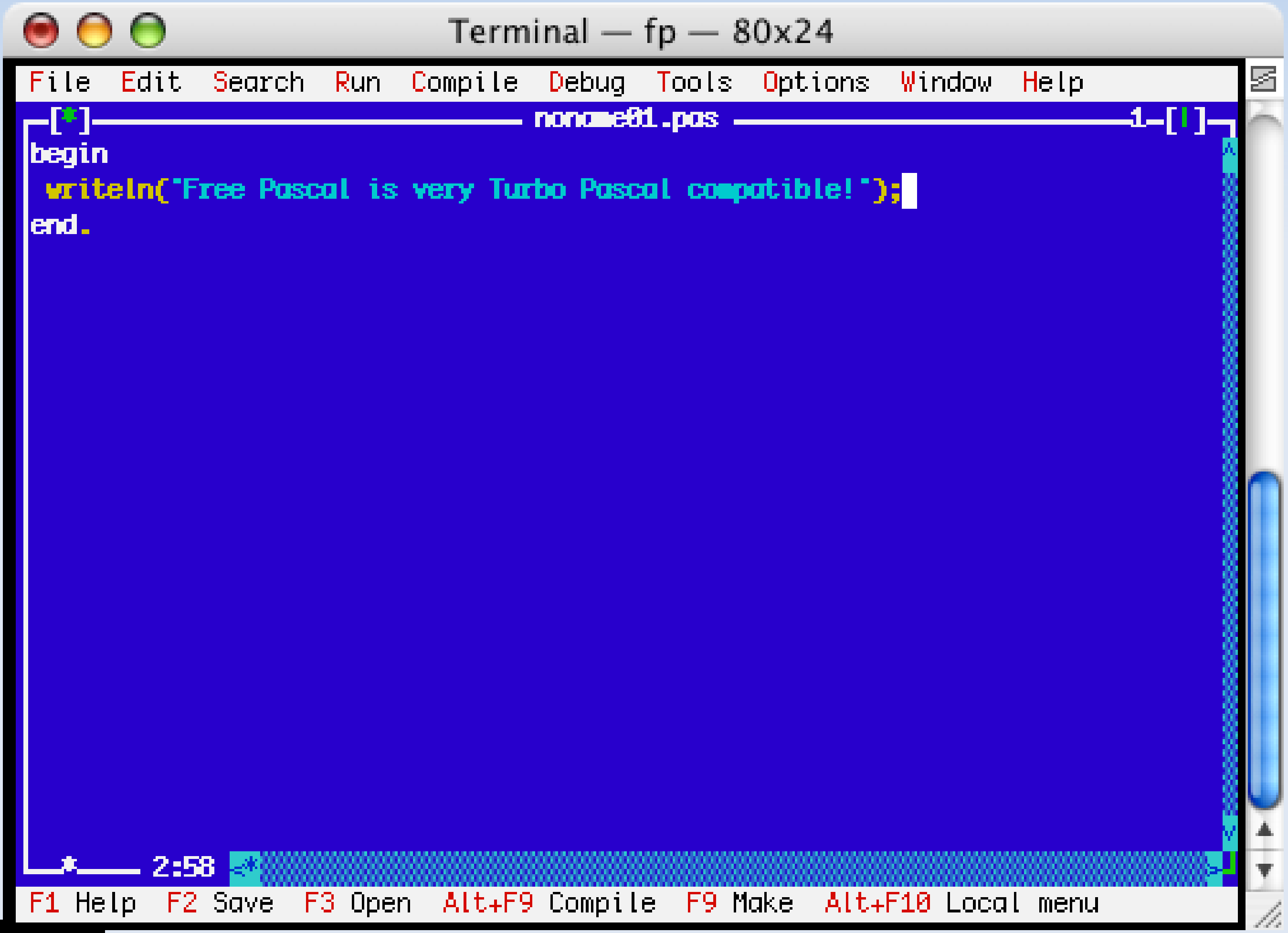
- My preferred solution is to create a config.json file and fetch() config data in Javascript file.
- **config.json**
- {
-   "apiKey": "My read-only API key"
- }
- **script.js**
- fetch('/config.json').then(function (config) {
-     console.log('API key:', config.apiKey);
- });

EKON 27

Demo: 1241_pixabay_api_tester.pas

Just a Shell

- IDE for Console or Terminal


Terminal — fp — 80x24
File Edit Search Run Compile Debug Tools Options Window Help
noname01.pas
2:58
F1 Help F2 Save F3 Open Alt+F9 Compile F9 Make Alt+F10 Local menu

```
begin
 writeln('Free Pascal is very Turbo Pascal compatible!');
end.
```

EKON 27

Pas2JS

- Pas2js is an open source Pascal to JavaScript transpiler. It parses Object Pascal or maXbox files and emits JScript. It takes Delphi/Lazarus projects and modules (.DPR, .LPR, .PAS, .PP) and converts them to JavaScript (.JS). The JS is currently of level ECMAScript 5 and should run in any browser or in Node.js (target “nodejs”). It is available in 5 forms:
- as a library
- as a command-line program
- as a webserver
- as a node.js program
- as a program running in the browser.

EKON 27

Using libpas2js.dll

- You can build libpas2js.dll directly by using Lazarus or lazbuild to compile
- compiler/utils/pas2js/pas2jslib.lpi

Anyway you should now have the library:

- libpas2js.so on Linux
- libpas2js.dylib on macOS
- libpas2js.dll on Windows

  It simply passes the command line parameters to run the compiler, so it behaves pretty much like the command line compiler **pas2js.exe.**

Let's compile

- It transpiles from actual Pascal source, it has no intermediate .ppu files. That means all sources must always be available.
- Const pas2jsPATH = 'C:\Program Files\Streaming\maxbox4\examples\pas2js-windows-2.2.0\pas2js-windows-2.2.0\bin\i386-win32\';
- writeln(GETDOSOutput('cmd.exe /c "'+
-                 pas2jsPATH+'pas2js" -Jc -Jirtl.js -Tbrowser ..\..\demo\chartjs\demoradar.lpr', Pas2jsPATH));
-
- Demo: *1238_create_process_etl_javascript.txt*

EKON 27

After Transpile

- Pas2JS Compiler version 2.2.0 [2022/02/22] for Win32 i386
- Copyright (c) 2021 Free Pascal team.
- C:\Program Files\Streaming\maxbox4\examples\pas2js-windows-2.2.0\pas2js-windows-2.2.0\demo\chartjs\demotime.lpr(9,3) Hint: Unit "Math" not used in demotime
- Info: 9627 lines in 7 files compiled, 0.1 secs
-
- mX4 executed: 11/08/2023 15:21:40 Runtime: 0:0:2.272 Memload: 54% use

EKON 27

Sign Compatability

- Simply add the config.json to your .gitignore and treat it the same as you would a .env file.
- Sign your script (yes we can, for windows)
- *TOSIGNFILE:= '1235_tetris_signed.js'*
- if fileExists(CERTFILE) then begin
-     writeln(botostr(ChDirW(TOOLPATH)));
-     passfromfile:= FileToString('./certs/passfile2.txt')
-     ExecuteShell('signtool.exe', 'sign /f'
-                 +' certs/maxbox4exe.pfx /p '+passfromfile
-                 +' /t http://timestamp.digicert.com '+TOSIGNFILE);

EKON 27

pas2js Electron Web App

- Install Electron and you must install node.js.
- Windows, MacOS: https://nodejs.org/en/download/
- Debian, Ubuntu:
- Check that node and npm work:
- node -v
- npm -v
- C:\box\mynodejs\node_modules\electron\dist\electron.exe

Class Definitions

Through external class definitions, the trans/compiler can use JavaScript classes:

- All classes available in the JavaScript runtime, and in the browser are available
- through import units (comparable to the windows or Unix units for the native compiler).
- For Node.js, basic support for the nodejs runtime environment is available.
- An import unit for jQuery is available (libjquery)
- a converter from maXbox to lpr project files

EKON 27

Distribution

- For the generated code to work, a small JavaScript file is needed: rtl.js. It defines an object rtl. This object will start the Object Pascal code if you include a call to rtl.run() in the HTML page. Then we pass the file to the transpiler:
- <script>
-     rtl.run();
- </script>
- pas2js can automatically include this file (rtl.js) in the generated output, like this:
- pas2js -Jc -Jirtl.js -Tbrowser demoradar.lpr

https://raw.githack.com/breitsch2/maXbox4/master/assets/demoradar.html

EKON 27

Content

- The pas2js compiler and RTL are – naturally – open source and can be downloaded and used freely. And I got my output as a javasscript file *demoradar.js*
- var pas = { $libimports: {}};
- var rtl = {
-   version: 20200,
-   quiet: false,
-   debug_load_units: false,
-   debug_rtti: false, $res : {},

EKON 27 Also: https://github.com/Kryuski/pas2js-for-delphi

HTML inline

```
▪ <!doctype html>
  <html lang="en">

    <head>
      <meta charset="utf-8">
      <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
      <meta name="description" content="Example showing how to use TchartJS">
      <meta name="author" content="silvioprog">
      <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.2.1/css/bootstrap.min.css"
  integrity="sha384-GJzZqFGwb1QTTN6wy59ffF1BuGJpLSa9DkKMp0DgiMDm4iYMj70gZWKYbI706tWS"
  crossorigin="anonymous">

      <script src="https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.7.3/Chart.min.js"  integrity="sha256-
  oSgtFCCmHWRPQ/JmR4OoZ3Xke1Pw4v50uh6pLcu+fIc=" crossorigin="anonymous"></script>
      <script src="./js/demoradar.js"></script>

▪     <title>TChartJS example</title>
      <style>
        .title {
          margin: 20px 0 20px 0
        }

        </style>
    </head>
```

EKON 27

view-source:https://raw.githack.com/breitsch2/maXbox4/master/assets/demoradar.html

Components

- Imageformats: .bmp, .png, .xpm, . jpg, .pnm, .tga (imagesforlazarus)
- OpenGL Components: lazopenglcontext (gtk, carbon, win32/64) oder glscene (linux/gtk, win32, SVG!, https://ideasawakened.com/post/simple-svg-images-in-delphi-applications)
- Internet/smtp/ftp/http/tcp: Synapse, Curl, Indy, Lnet, TRestclient
- Code-Formater: prettyformat, Charts, Bootstrap
- ...

https://raw.githack.com/breitsch2/maXbox4/master/assets/graph3.html

EKON 27

Project Examples

https://www.clevercomponents.com/articles/article052/

MRIcron

ARACHNE

TRUCK BYTES

https://linuxschweizag.wordpress.com/2023/04/06/tutorials/

OutKafe

master maths
make your mark

DEDALU

https://maxbox4.wordpress.com/2023/05/23/mapbox-in-maxbox/

Becape

SatuVISI Indict

Audio X

https://raw.githack.com/breitsch2/maXbox4/master/assets/pacman2/pacman.html

Siages

Cactus Jukebox

EKON 27

https://youtu.be/SC3i7Ru8XPY

Pas2JS & JS1Pas

Thanks for coming!

Materials:

https://github.com/breitsch2/maXbox4/tree/master/assets

EKON 27