

PureScript

Jemný úvod

Matej Fandl, 10. 5. 2021

Agenda

1. Haskell - kam teraz?
2. Haskell a webové prehliadače
3. PureScript
 - a. čo
 - b. prečo
 - c. ako

Haskell - kam teraz?

Pandoc

If you need to convert files from one markup format into another, pandoc is your swiss-army knife.

<https://pandoc.org/>

Hakyll

Hakyll is a Haskell library for generating static sites, mostly aimed at small-to-medium sites and personal blogs. It is written in a very configurable way and uses an xmonad-like DSL for configuration.

<https://jaspervdj.be/hakyll/>

Copilot

stream-based runtime-verification
framework for generating hard
real-time C code.

<https://copilot-language.github.io/index.html>

Dhall

Dhall is a programmable
configuration language that you can
think of as: JSON + functions +
types + imports

<https://dhall-lang.org/>

LambdaCube

LambdaCube 3D is Haskell-like
purely functional domain specific
language for programming the GPU
(graphics processing unit).

<http://lambdacube3d.com/>

Diagrams

Diagrams is a full-featured framework and embedded domain-specific language for creating declarative vector graphics and animations.

<https://archives.haskell.org/projects.haskell.org/diagrams/>

Accelerate

Accelerate is a language for array-based computations, designed to exploit massive parallelism.

<https://www.acceleratehs.org/>

Hasura

Instant GraphQL for all your data

<https://hasura.io/>

ShellCheck

finds bugs in your shell scripts

<https://www.shellcheck.net/>

Web Development

<https://www.spock.li/>

<https://github.com/scotty-web/scotty>

<https://www.yesodweb.com/>

<https://docs.servant.dev/en/stable/>
(api's)

Client Side

[The JavaScript Problem](#)

JavaScript problem

```
"b" + "a" + +"a" + "a"; // -> 'baNaNaN'
```

```
parseInt(0.000001); // -> 0
```

```
parseInt(0.0000001); // -> 1
```

```
true + true; // -> 2
```

```
' ' + ' ' // -> ' '
```

```
[] + [] // -> ' '
```

```
{ } + [ ] // -> 0
```

```
[] + { } // -> '[object Object]'
```

```
{ } + { } // -> '[object Object][object Object]'
```

JavaScript pro

```
"b" + "a" + +"a" + "a"
```

```
parseInt(0.000001);
```

```
parseInt(0.0000001)
```

```
true + true; // ->
```

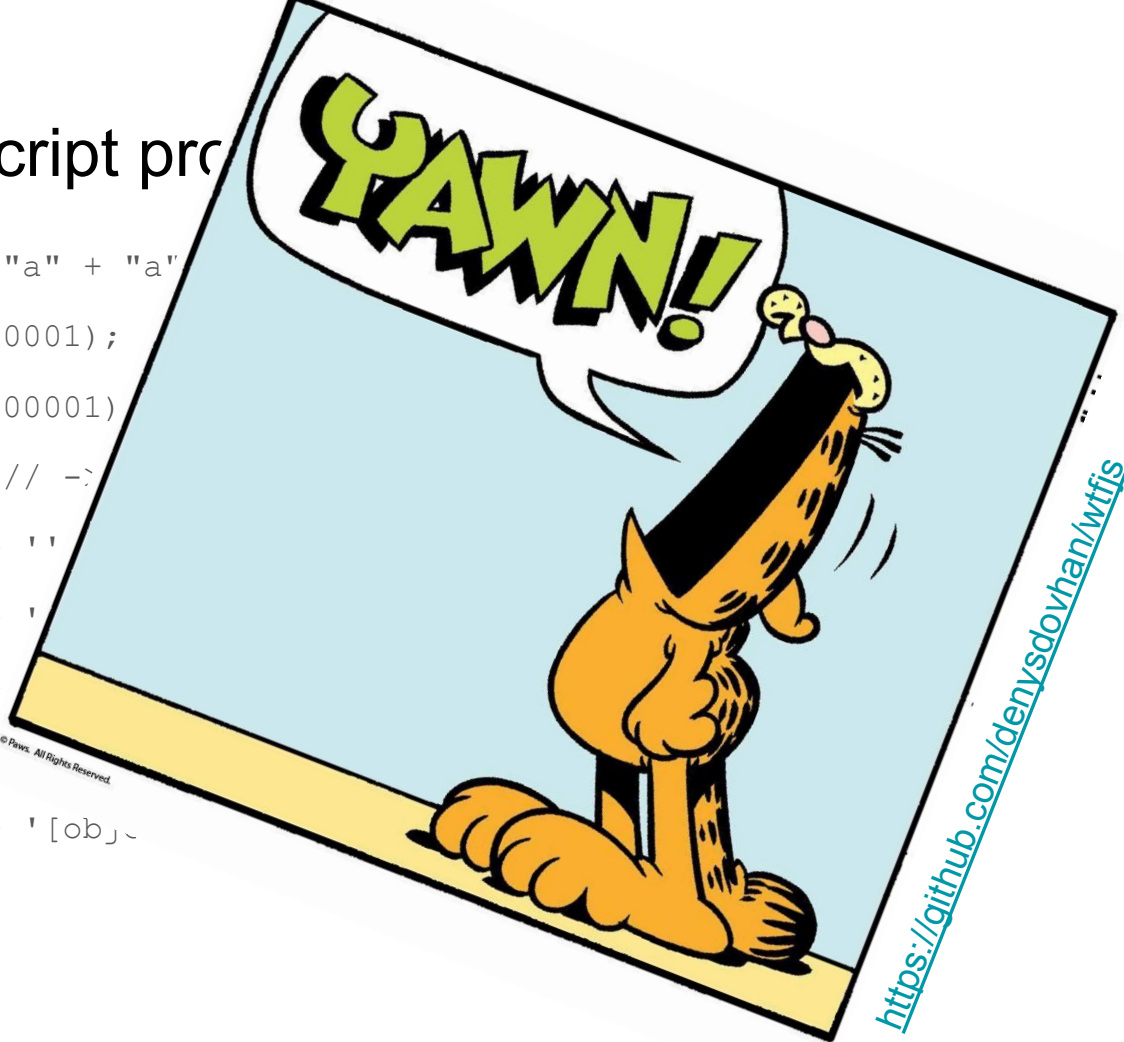
```
' ' + ' ' // -> ' '
```

```
[] + [] // -> ' '
```

```
{ } + [ ] // ->
```

```
[ ] + { } // -
```

```
{ } + { } // -> '[obj]~'
```



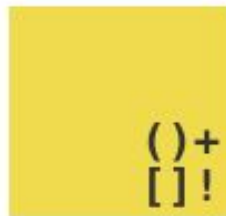
© Paws. All Rights Reserved.

<https://github.com/denysdovhan/wtfjs>

JSFuck

JSFuck is an esoteric and educational programming style based on the atomic parts of JavaScript. It uses only six different characters to write and execute code.

<http://www.jsfuck.com/>



○ ○ ○

```
1 const allowList = [1,2,3]
```

```
2 const elements = [5, 7, 9, 1, 4, 3]
```

```
3
```

```
4 elements.filter(x => allowList.includes(x)) // → [1, 3]
```

```
5 elements.filter(Array.prototype.includes.bind(allowList)) // → []
```

Is it ty, Haskell?

```
import Prelude
import Effect.Console (log)

greet :: String -> String
greet name = "Hello, " <> name
<> "!"

main = log (greet "World")
```

GHCJS

GHCJS is a Haskell to JavaScript compiler that uses the GHC API.

Starting with GHC version 8.2, GHCJS depends on a customized ghc library, installed under the name ghc-api-ghcjs

<https://github.com/ghcjs/ghcjs>



<https://haskell-miso.org/>

Asterius

Asterius is a Haskell to WebAssembly compiler based on GHC. It compiles Haskell source files or Cabal executable targets to WebAssembly+JavaScript code which can be run in Node.js or browsers.

<https://github.com/tweag/asterius>



PureScript

A strongly-typed functional programming language that compiles to JavaScript

<https://www.purescript.org/>



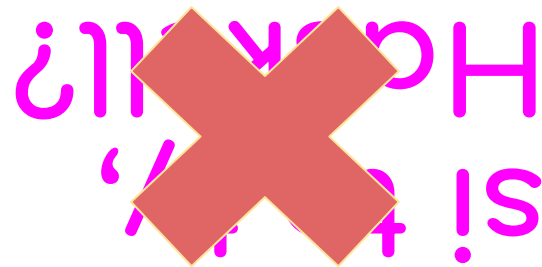
```
import Prelude

import Effect.Console (log)

greet :: String -> String

greet name = "Hello, " <> name
<> "!"

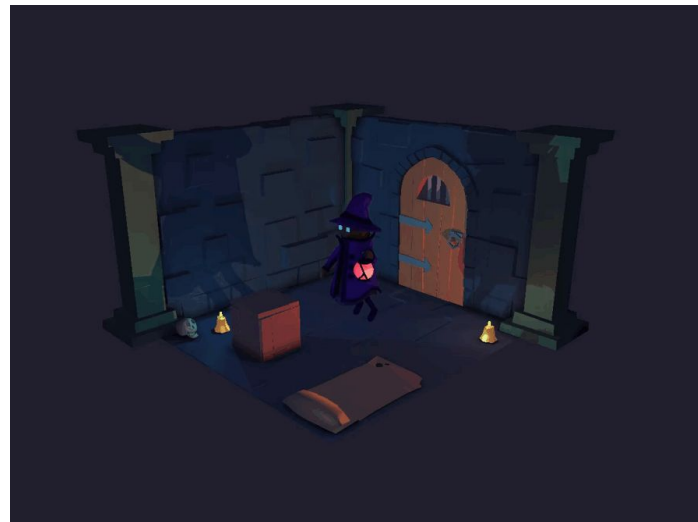
main = log (greet "World")
```



PureScript

- kompilácia do js (primárne)
- interop s js
- čitateľný js kód

- purescript-native - c++, go



purescript-native + Panda3D

<https://github.com/lettier/lambda-lantern>


PureScript - tooling

- purs - kompilátor - <https://www.purescript.org/>
- spago - package manager, build tool - <https://github.com/purescript/spago>
- editor support - Language Server Protocol + <https://github.com/nwolverson/purescript-language-server>, čiže všetky moderné editory ((neo)vim, emacs, atom, vscode)

PureScript v skratke

<https://github.com/purescript/documentation/blob/master/language/Differences-from-Haskell.md>

striktné vyhodnocovanie

take 10 [1..] 

predikovateľný výkon 

interop s JS



Prelude

iba modul, nič implicitné

FFI

```
foreign import jsFunkcia :: Typ -> Typ
```

js funkcia je v *foreign JavaScript module*
Nazov.purs -> Nazov.js

Tuple nie je (,)

ale `Data.Tuple` z *purescript-tuples*

[] nie je list

ale typ Array - js pole / v type signature napr `nums :: Array Int`

chcete list?

```
spago install purescript-lists
import Data.List
import Data.List.Lazy
```


IO -> Effect

natívne side efekty

efekt je funkcia bez argumentov

<https://github.com/purescript/purescript-effect/tree/master/src>

forall

explicitná deklarácia typových premenných v polymorfných funkciách

```
foo :: forall a b. a -> b
```

Records

JS objekt

bodka

```
type Osoba = { meno :: String, vek :: Int }  
osoba :: Osoba  
osoba = { meno: "Meno", vek: 94 }  
osoba.meno  
osoba.vek
```

Row Polymorphism

```
showVek :: Osoba -> String  
showVek :: forall r. { vek :: Int | r } ->  
String  
showVek rec = show rec.vek  
  
showVek' { vek } = show vek
```

Kompozícia <<< a >>>

bodku už máme v Recordoch

derivovanie inštancií typeclassov

~~deriving (Eq, Ord)~~

```
data Foo = Foo Int String
```

```
derive instance eqFoo :: Eq Foo
```

```
derive instance ordFoo :: Ord Foo
```

d'akujem za pozornosť

matej.fandl@fmph.uniba.sk