

# Plugin Architectures in Haskell

An Overview over the ecosystem

---

Sebastian Graf

September 15, 2016

<https://github.com/sgraf812/hal16>

# Motivation

---

# Problem Description

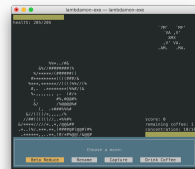


$(\lambda x.xx)(\lambda x.xx)$



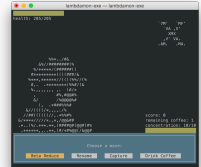
[1] [2]

# Plugin Architecture Requirements



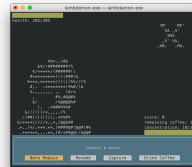
## Plugin Architecture Requirements

## Extensibility through third party code



# Plugin Architecture Requirements

Extensibility through third party code  
Haskell as extension language

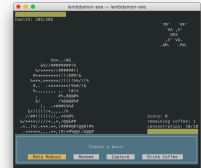


# Plugin Architecture Requirements

**Extensibility** through third party code

**Haskell** as extension language

**Stand-alone** No compiler toolchain should be required on the client



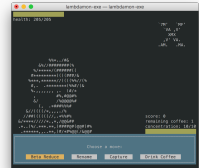
# Plugin Architecture Requirements

**Extensibility** through third party code

**Haskell** as extension language

**Stand-alone** No compiler toolchain should be required on the client

**Type safety** Early and graceful recognition of incompatible extensions





# Plugin Architecture Requirements

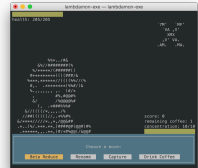
**Extensibility** through third party code

**Haskell** as extension language

**Stand-alone** No compiler toolchain should be required on the client

**Type safety** Early and graceful recognition of incompatible extensions

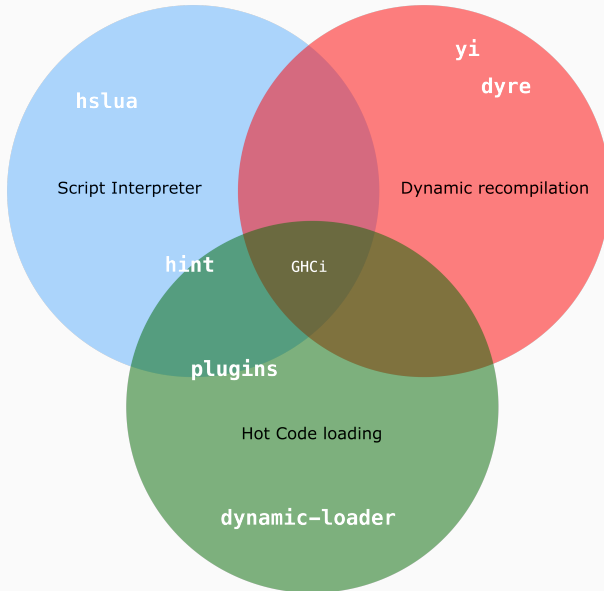
**Maturity** Easy integration in a **Cabal** build process



# Shootout

---

# Contenders



**Extensibility** can't be easier for third parties, see WoW. ✓✓

**Haskell** is not lua. ✗

**Stand-alone** The C bits are statically linked, no further dependencies. ✓✓

**Type safety** Neither in called code nor at API boundaries, also lua stack. ✗✗

**Maturity** lua is battle-tested and dead simple to include, yet hslua's API is rather low-level. ✓



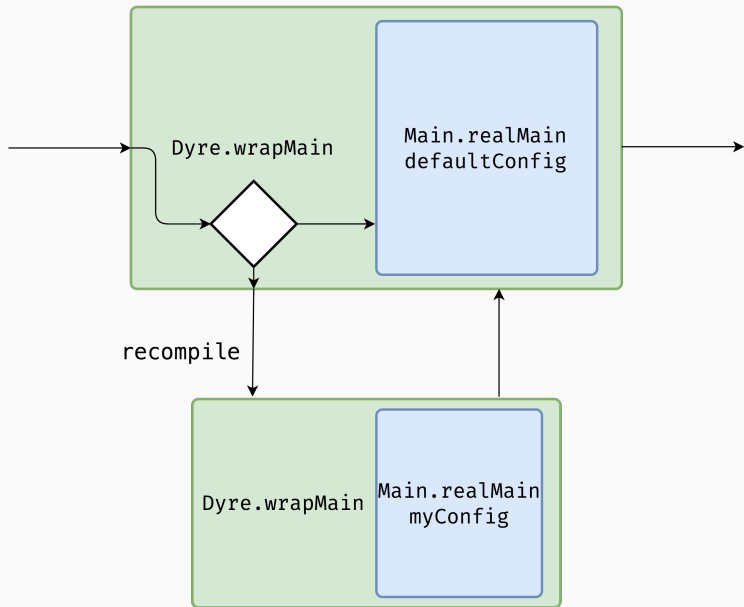
**Extensibility** Just drop in source files, although package dependencies are resolved through GHC package registry ✓

**Haskell** ✓

**Stand-alone** Uses the GHC API, including compilation specific settings paths ✗

**Type safety** through broken **Typeable** overloads, falling back to **read/show** serialization. ✗

**Maturity** Most-used (52 reverse deps) contender according to hackage. ✓





**Extensibility** You can't have more than one config file. Merging them requires knowledge of Haskell. ✗

**Haskell** ✓

**Stand-alone** Needs a complete compiler/stack toolchain available. ✗✗

**Type safety** There are no API boundaries, it's all one program and consequently type-checked as one. ✓✓

**Maturity** Rotting. Only really works with GHC, no cabal or stack. Mind-bending API. ✗

**Extensibility** Just drop in object archives. ✓✓

**Haskell** ✓

**Stand-alone** Although it depends on the GHC API, it works on a fresh installation. ✓

**Type safety** Needs reproducible builds in order to work seamlessly. Installed package id needed to find objects. Type errors at API boundaries lead to crashes at runtime. ✗

**Maturity** Unwieldy, scarcely documented API. Handling GHC generated symbols is low-level and unresolved. 0 reverse deps. ✗



# A word about plugins



**Extensibility** Just drop in object files. Package dependencies via `package.conf`s (outdated) ✓✓

Haskell ✓

**Maturity** Nicer API than `dynamic-loader`, but it's horribly outdated and broken. ~~XXX~~

# Summary

	hslua	hint	dyre	dynamic-loader	plugins
Extensibility	✓✓	✓	✗	✓✓	✓✓
Haskell	✗	✓	✓	✓	✓
Stand-alone	✓✓	✗	✗✗	✓	?
Type safety	✗✗	✗	✓✓	✗	?
Maturity	✓	✓	✗✗	✗	broken

Thanks! Questions?

# References



<http://www.ebay.com/itm/Anime-Cosplay-Pokemon-Go-Pocket-Monster-Ash-Ketchum-232012326919>.

Accessed: 2016-09-09.



[https://upload.wikimedia.org/wikipedia/commons/1/17/Rogue\\_Screen\\_Shot\\_CAR.PNG](https://upload.wikimedia.org/wikipedia/commons/1/17/Rogue_Screen_Shot_CAR.PNG).

Accessed: 2016-09-09.

Check out the code of this talk at <https://github.com/sgraf812/hal16>