ToiletPaper #78

jambit
WHERE INNOVATION WORKS

Elm

From Andreas Scharf

✖ Problem

You need a **statically typed pure functional programming language** for web development.

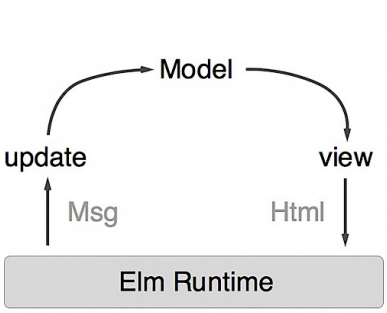
✔ Solution

Evan Czaplicki (*@evancz*) started to develop **Elm** as his master thesis in 2012. Elm is a functional programming language with the goal to avoid side effects and focus on simplicity and performance. The code is compiled to JavaScript. Elm comes with REPL, package manager, time-traveling debugger, development server, and quality tooling. Due to static type checking the project is promising to have no runtime exceptions in practice. The compiler is known to give useful hints, infer types, and return guesses for spelling mistakes. All values in Elm are immutable. The language has its own, rather small module ecosystem which is enforcing semantic versioning based on library interfaces. JavaScript interoperability in both directions is possible.

➔ Example

The **Elm Architecture** is quite popular and its design has influenced the predictable state container Redux. The example code is rendering the user's Markdown content to HTML and will demonstrate the typical Elm application structure with Model, View and Update.

- The **Model** is defining a record containing the application state.
- The **View** is a function taking the Model as input and returning HTML markup, representing the application state in a declarative way.
- The **Update** function receives the current Model and a Message and returns a new, updated Model.



The resulting application is a continuous loop. New user input results in a change message with content. The update function replaces the model content. The new model is passed to the view function and rendered to HTML using a Markdown library.

```elm
main : Program () Model Msg
main = Browser.sandbox { init = initialModel , view = view , update = update }

-- MODEL
type alias Model = { content : String }

initialModel : Model
initialModel = { content = "# Headline" }

-- UPDATE
type Msg = Change String

update : Msg -> Model -> Model
update msg model =
    case msg of
    |   Change content -> { model | content = content }

-- VIEW

view : Model -> Html Msg
view model =
    div []
    |   [ textarea [ placeholder "Markdown", onInput Change, value model.content ] []
    |   , Markdown.toHtml [] model.content
    |   ]
```

Try out the online version of the example code under the following link: *https://ellie-app.com/43JYBXN4DQ9a1*

+ Links

- Project: http://www.elm-lang.org
- Guide: https://guide.elm-lang.org
- Try ELM online: https://ellie-app.com/new
- Curated list: https://github.com/isRuslan/awesome-elm
- Elm architecture: https://dennisreimann.de/articles/elm-architecture-overview.html