Update from the Elixir team

elixir-lang.org

Elixir v1.0

Sep 2014

> 180 contributors

Elixir v1.4

Jan 2017

> 520 contributors

Release every 6 months

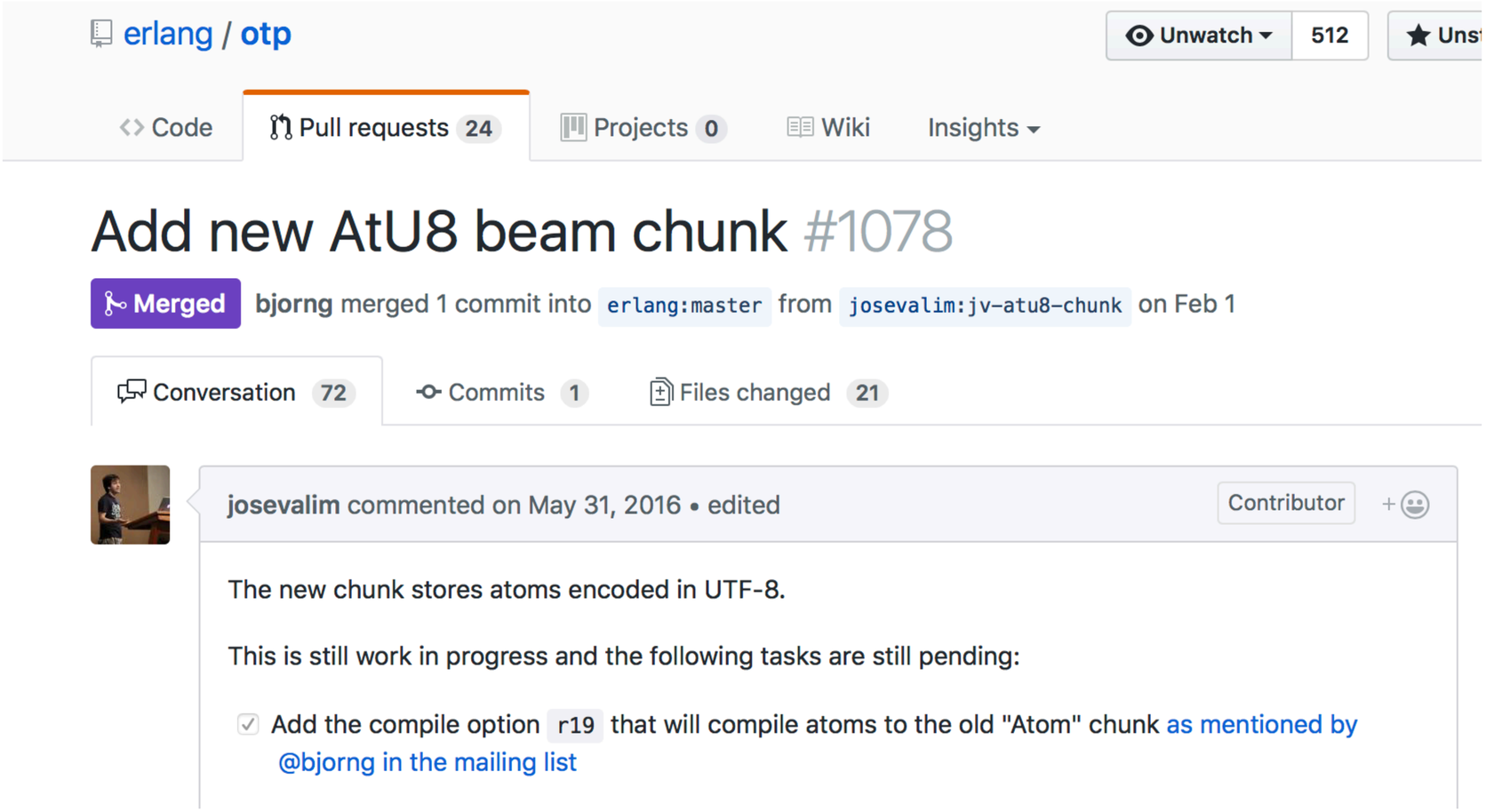
v1.5 → mid 2017
v1.6 → beg 2018
… → …

IMPROVEMENTS IN v1.5

UTF-8 atoms

:"  "

Contributed back to OTP



erlang / otp

Unwatch 512 Unst

Code | Pull requests 24 | Projects 0 | Wiki | Insights

Add new AtU8 beam chunk #1078

Merged **bjorng** merged 1 commit into `erlang:master` from `josevalim:jv-atu8-chunk` on Feb 1

Conversation 72 | Commits 1 | Files changed 21

josevalim commented on May 31, 2016 • edited — Contributor

The new chunk stores atoms encoded in UTF-8.

This is still work in progress and the following tasks are still pending:

- [x] Add the compile option `r19` that will compile atoms to the old "Atom" chunk as mentioned by @bjorng in the mailing list

```
test "こんにちは世界" do
  assert :こんにちは世界
end
```

UTF-8 VARIABLES

```
josé = "wat"
josé
#=> "wat"
```

http://erlang.org/pipermail/erlang-questions/2012-October/069820.html

Exception.blame/3

```
iex(1)> Access.fetch(:foo, :bar)
** (FunctionClauseError) no function clause matching in Access.fetch/2

    The following arguments were given to Access.fetch/2:

        # 1
        :foo

        # 2
        :bar

    Attempted function clauses (showing 5 out of 5):

        def fetch(%struct{} = container, key)
        def fetch(map, key) when is_map(map)
        def fetch(list, key) when is_list(list) and is_atom(key)
        def fetch(list, key) when is_list(list)
        def fetch(nil, _key)

    (elixir) lib/access.ex:261: Access.fetch/2
```

```
def fetch(%struct{} = container, key)
def fetch(map, key) when is_map(map)
def fetch(list, key) when is_list(list) and is_atom(key)
def fetch(list, key) when is_list(list)
def fetch(nil, _key)
```

FunctionClauseError at GET /oops

no function clause matching in Access.fetch/2

lib/elixir/lib/access.ex

```
256     """
257     @spec fetch(container, term) :: {:ok, term} | :error
258     @spec fetch(nil_container, any) :: :error
259     def fetch(container, key)
260
261     def fetch(%struct{} = container, key) do
262       struct.fetch(container, key)
263     rescue
264       e in UndefinedFunctionError ->
265         raise_undefined_behaviour e, struct, {^struct, :fetch, [
266     end
```

elixir · Access.fetch/2 (docs)

▶ Called with 2 arguments

▼ Available clauses (5 out of 5)

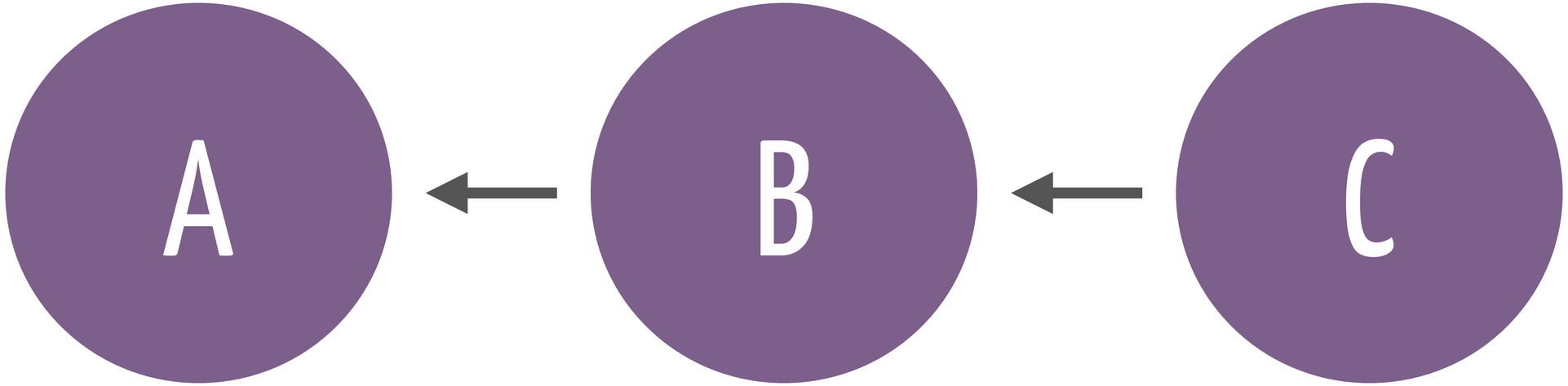
- `def fetch(%struct{} = container, key)`
- `def fetch(map, key) when is_map(map)`
- `def fetch(list, key) when is_list(list) and is_atom(key)`
- `def fetch(list, key) when is_list(list)`
- `def fetch(nil, _key)`

☑ Show all frames

elixir · lib/elixir/lib/access.ex:261	Access.fetch/2
foo.exs:16	anonymous fn/1 in MyRouter.do_match/4
foo.exs:2	MyRouter.plug_builder_call/2
lib/plug/debugger.ex:100	MyRouter.call/2
plug · lib/plug/adapters/cowboy/handler.ex:15	Plug.Adapters.Cowboy.Handler.upgrade/4
cowboy · src/cowboy_protocol.erl:442	:cowboy_protocol.execute/4

GenStage + Flow

GenStage



stages of computation with backpressure

Flow

parallel computations on collections

```
File.stream!("path/to/some/file")
|> Flow.from_enumerable()
|> Flow.flat_map(&String.split(&1, " "))
|> Flow.partition()
|> Flow.reduce(fn -> %{} end, fn word, acc ->
  Map.update(acc, word, 1, &(&1 + 1))
end)
|> Enum.to_list()
```

elixir-lang / **gen_stage**

elixir-lang / **flow**

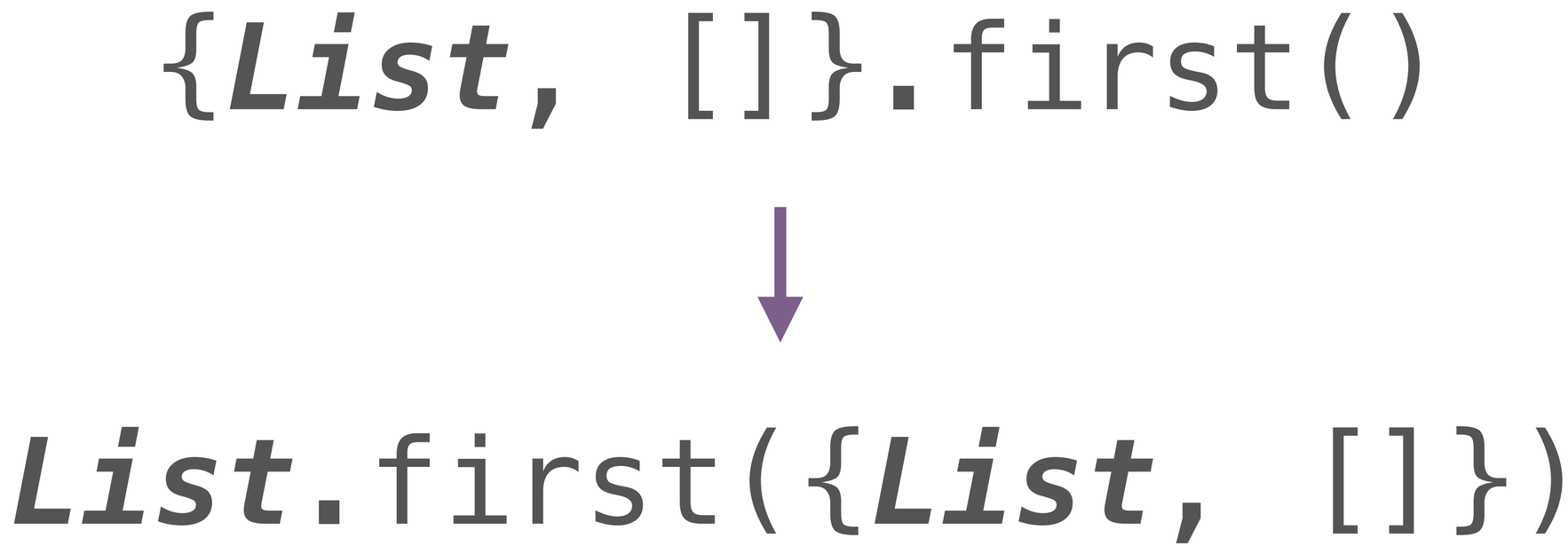
http://bit.ly/genstage

Many small improvements

CHANGELOG.md

FUTURE

Deprecate tuple calls


{List, []}.first()
List.first({List, []})

Dynamic supervisor

`simple_one_for_one`

confusing/mixed API

documentation

DynamicSupervisor

simple_one_for_one

confusing/mixed API

documentation

RESEARCH PROJECTS

Data streams
+
Property testing

```
test "starts_with?/2" do
  assert for s1 <- Data.string(),
              s2 <- Data.string() do
    String.starts_with?(s1 <> s2, s1)
  end
end
```

Google Summer of Code

ExFormat

```
def  foo (
 a, b ) do a+ b
    end
```

```
def foo(a, b) do
  a + b
end
```

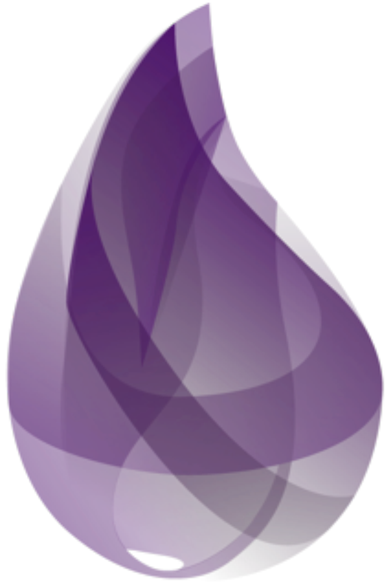
Language Server Protocol

http://langserver.org

GenHTTP

Functional, flexible HTTP client

github.com/elixir-lang/elixir



elixir

elixir-lang.org