Algebraic Effects and Static Analysis

for Safety-Critical Applications in Fuzion

Fridtjof Siebert, Michael Lill, Max Teufel
Tokiwa Software GmbH

AEiC 2024, 12. Jun 2024, Barcelona

TOKIWA
software

Overview

- ➡ Fuzion quick intro
- ➡ Algebraic Effects in Fuzion
- ➡ Toolchain and Static Analysis
- ➡ Safety-Critical Aspects
- ➡ Status

Overview

- ➡ Fuzion quick intro
- ➡ Algebraic Effects in Fuzion
- ➡ Toolchain and Static Analysis
- ➡ Safety-Critical Aspects
- ➡ Status

Fuzion Intro

➞ One concept: a feature

- takes role of class / method / procedure / etc.

➞ Fuzion is

- statically typed
- polymorphic: union types, parametric types, inheritance
- pure using effects

Overview

- → Fuzion quick intro
- → Algebraic Effects in Fuzion
- → Toolchain and Static Analysis
- → Safety-Critical Aspects
- → Status

Algebraic Effects

Definition

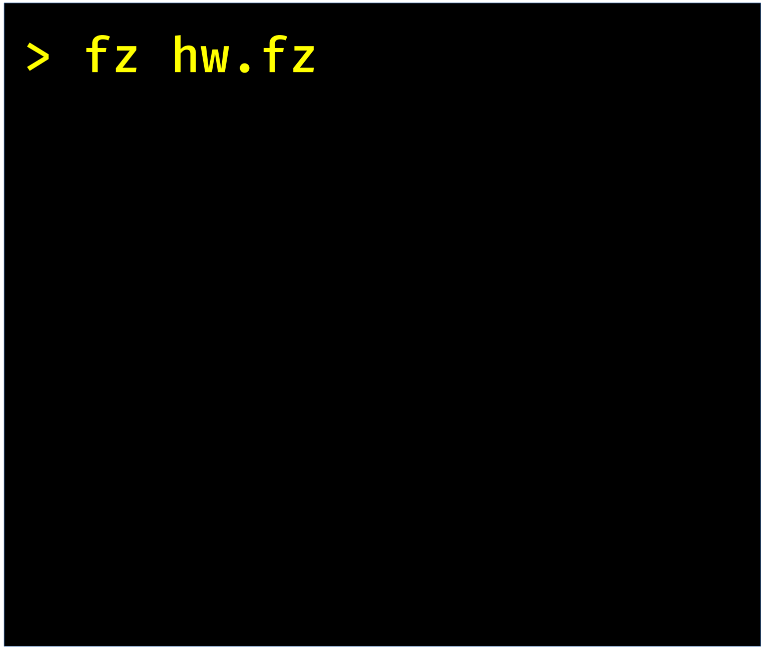
- ➡ an algebraic effect is a set of operations
  - `read`, `get_time`, `panic`, `log`, ...
  - operations often model a non-functional aspect
- ➡ operations may **resume** or **abort**
- ➡ an effect is implemented by a **handler**
- ➡ to use effect operations, a handler must be installed

Fuzion Effects Example

```
hello_world ! io.out ⇒
  io.out.println "hello world!"

hello_world
```

Fuzion Effects Example

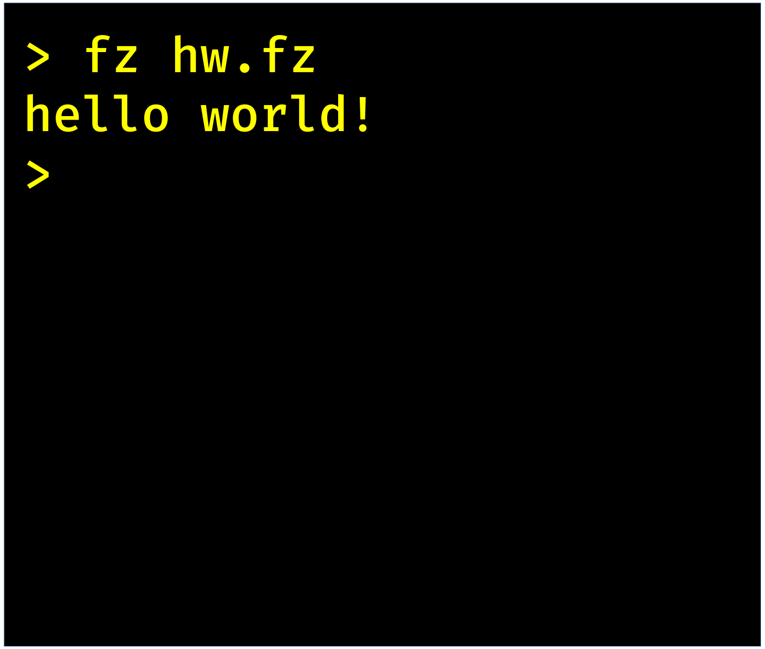
```
hello_world ! io.out ⇒
  io.out.println "hello world!"

hello_world
```



```
> fz hw.fz
```

Fuzion Effects Example

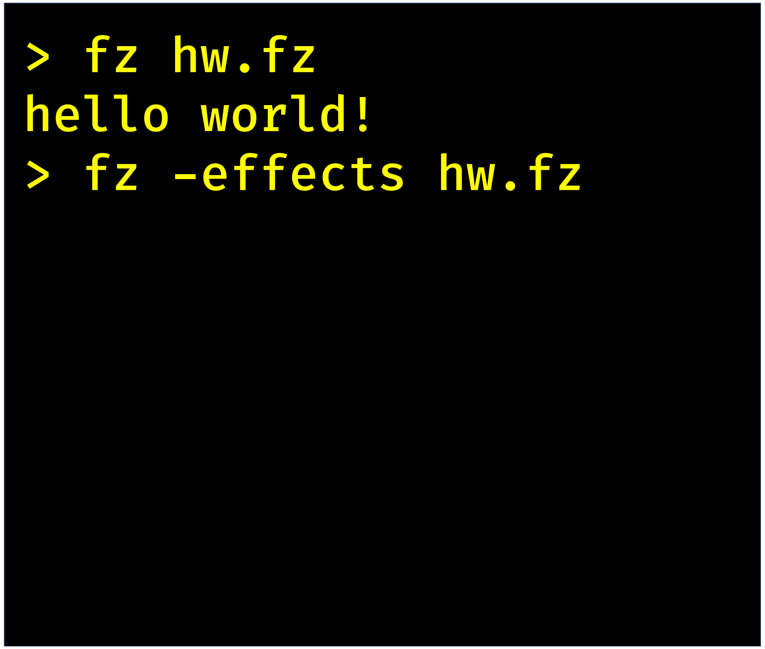
```
hello_world ! io.out ⇒
  io.out.println "hello world!"

hello_world
```



```
> fz hw.fz
hello world!
>
```

Fuzion Effects Example

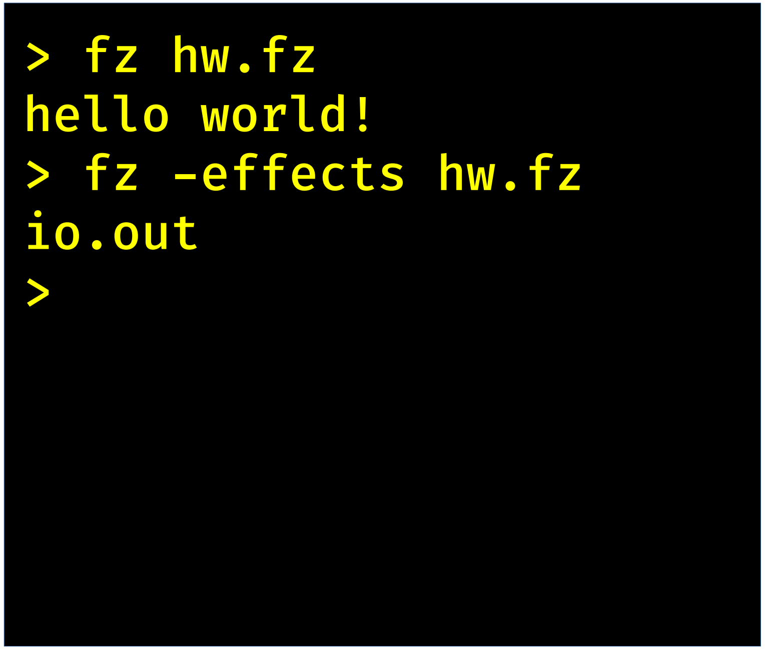
```
hello_world ! io.out ⇒
  io.out.println "hello world!"

hello_world
```



```
> fz hw.fz
hello world!
> fz -effects hw.fz
```

Fuzion Effects Example

```
hello_world ! io.out =>
  io.out.println "hello world!"

hello_world
```



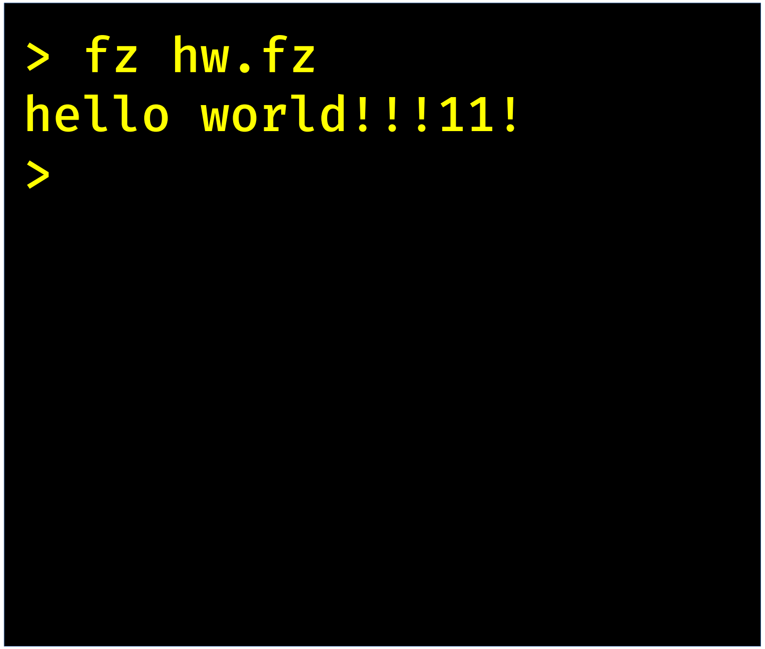
```
> fz hw.fz
hello world!
> fz -effects hw.fz
io.out
>
```

Fuzion Effects Example

```
hello_world ! io.out ⇒
  io.out.println "hello world!"

my_handler : io.Print_Handler is
  print(s Any) ⇒
    io.err.print (($s).replace "!" "!!!11!")

(io.out my_handler).go ()→
  hello_world
```

Fuzion Effects Example

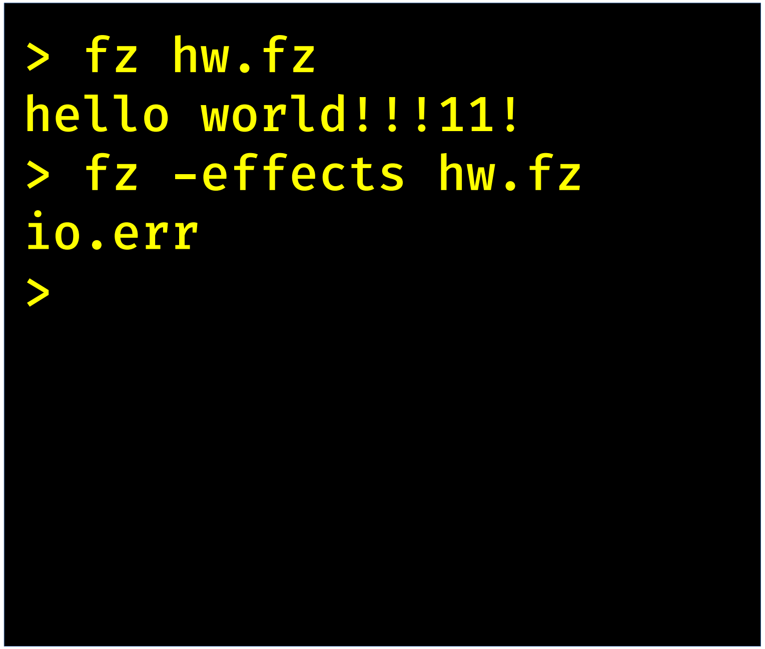
```
hello_world ! io.out ⇒
  io.out.println "hello world!"

my_handler : io.Print_Handler is
  print(s Any) ⇒
    io.err.print (($s).replace "!" "!!!11!")

(io.out my_handler).go ()→
  hello_world
```



```
> fz hw.fz
hello world!!!11!
>
```

Fuzion Effects Example

```
hello_world ! io.out ⇒
  io.out.println "hello world!"

my_handler : io.Print_Handler is
  print(s Any) ⇒
    io.err.print (($s).replace "!" "!!!11!")

(io.out my_handler).go ()→
  hello_world
```



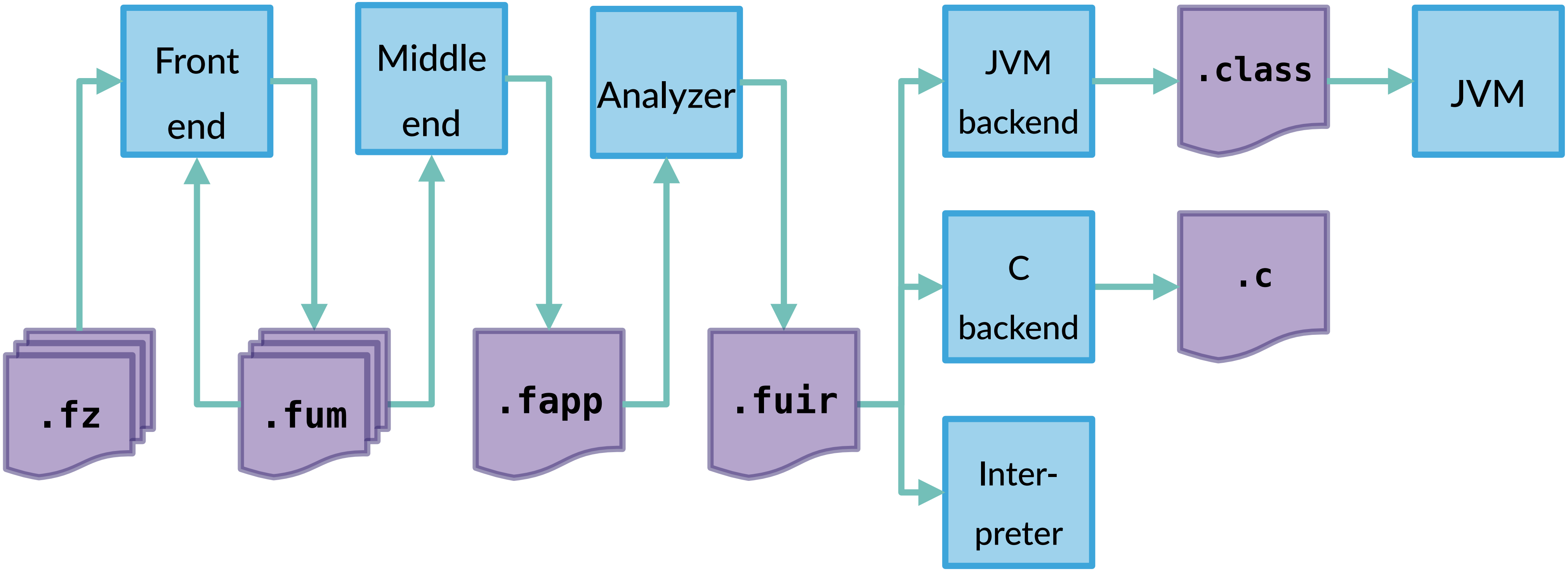
```
> fz hw.fz
hello world!!!11!
> fz -effects hw.fz
io.err
>
```

TOKIWA software

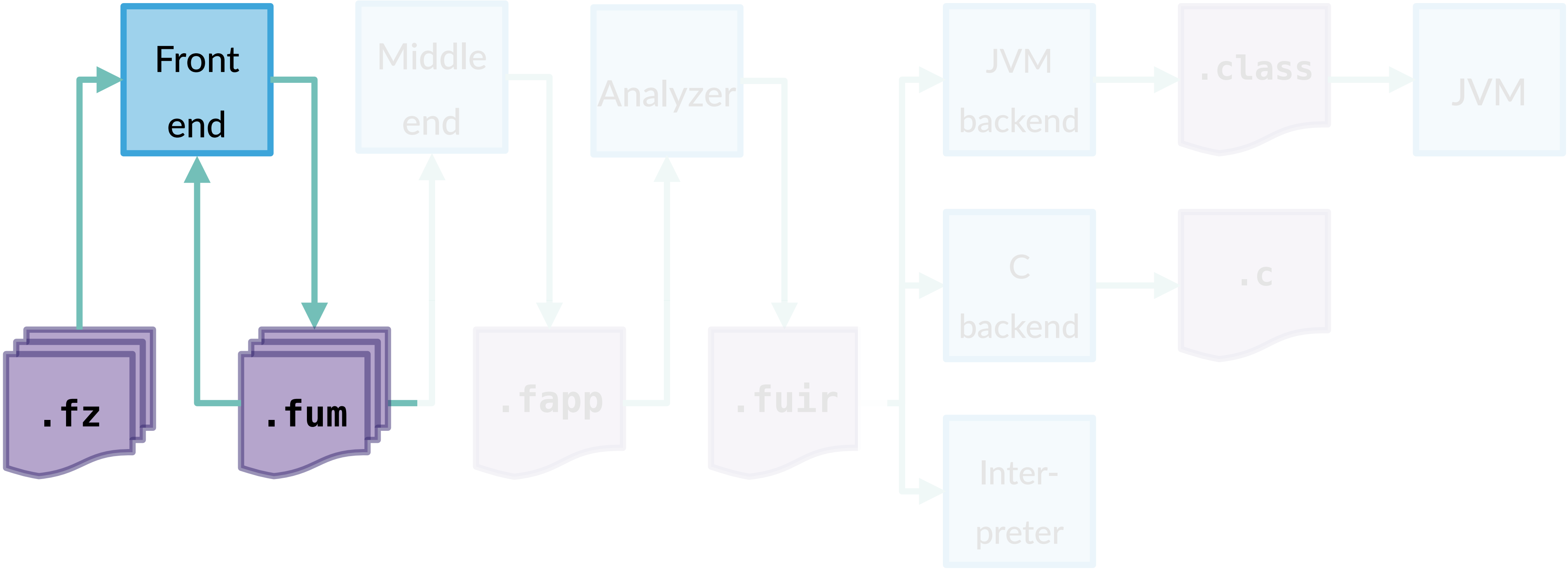
Overview

- ➡ Fuzion quick intro
- ➡ Algebraic Effects in Fuzion
- ➡ **Toolchain and Static Analysis**
- ➡ Safety-Critical Aspects
- ➡ Status

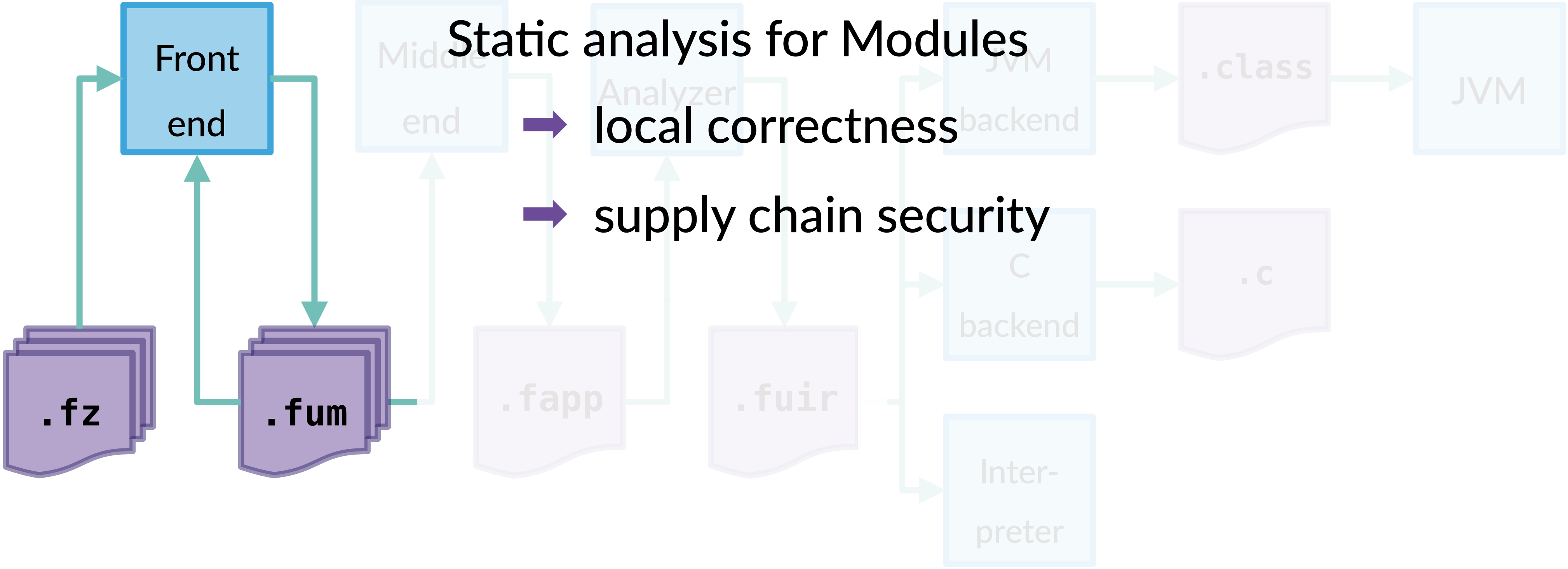
Fuzion Toolchain and Static Analysis



Fuzion Toolchain and Static Analysis



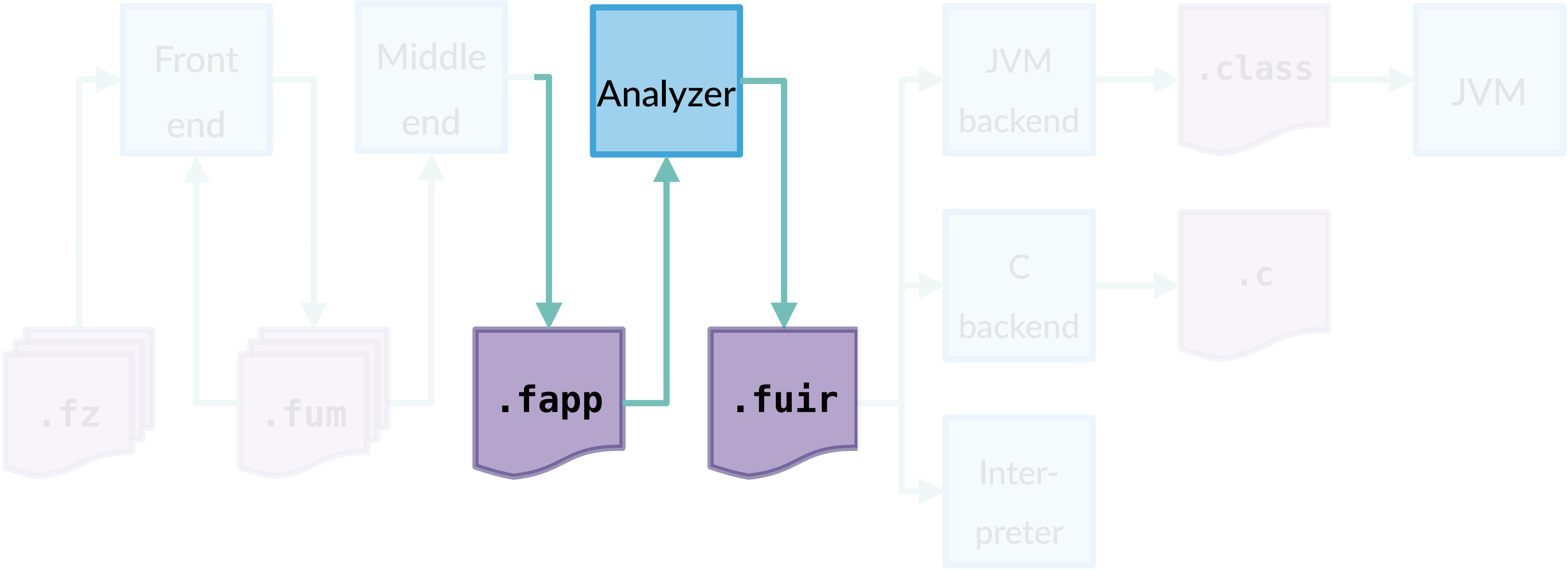
Fuzion Toolchain and Static Analysis



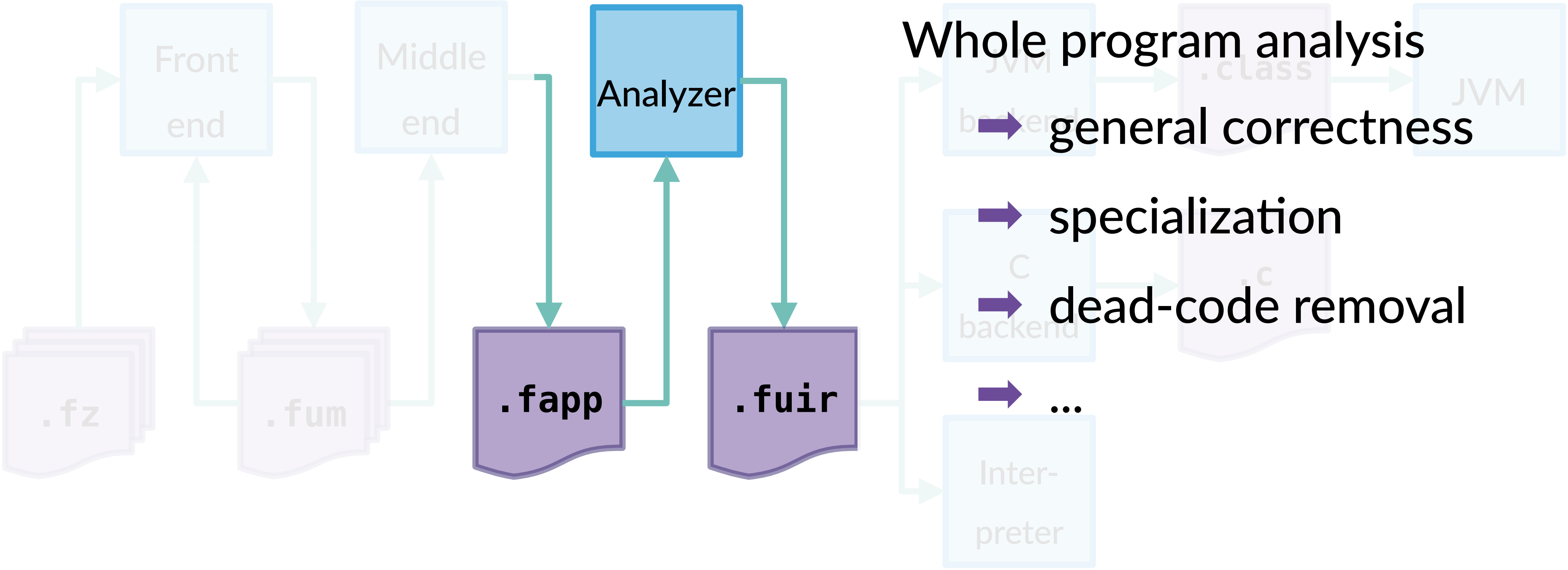
Static analysis for Modules

- ➡ local correctness
- ➡ supply chain security

Fuzion Toolchain and Static Analysis



Fuzion Toolchain and Static Analysis



Whole program analysis

- ➡ general correctness
- ➡ specialization
- ➡ dead-code removal
- ➡ ...

Overview

- ➡ Fuzion quick intro
- ➡ Algebraic Effects in Fuzion
- ➡ Toolchain and Static Analysis
- ➡ Safety-Critical Aspects
- ➡ Status

Effects for Safety-Critical Aspects

Use effect environments to guide static analysis

- ➡ Time requirements: constant / bounded / wcet
- ➡ non-blocking code
- ➡ interruptible code / timeout
- ➡ no heap allocation

Overview

- → Fuzion quick intro
- → Algebraic Effects in Fuzion
- → Toolchain and Static Analysis
- → Safety-Critical Aspects
- → Status

Fuzion: Status

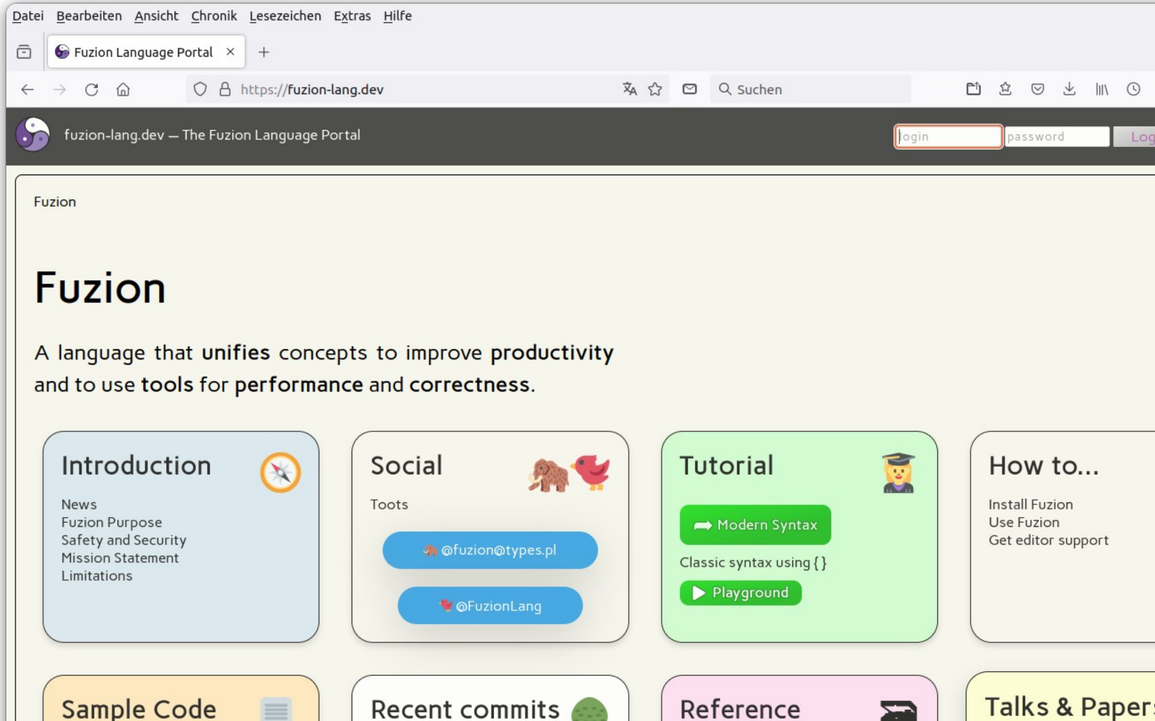
Fuzion still under development

- ➡ language definition slowly getting stable
- ➡ base library work in progress
- ➡ current implementation providing JVM and C backends
- ➡ Basic analysis tools available

Thank you. Any questions?

Please follow and stay informed

- ➡ https://github.com/tokiwa-software/fuzion
- ➡ https://fuzion-lang.dev
- ➡ @FuzionLang
- ➡ @Fuzion@types.pl



Fuzion

Fuzion

A language that **unifies** concepts to improve **productivity** and to use **tools** for **performance** and **correctness**.

Introduction

News
Fuzion Purpose
Safety and Security
Mission Statement
Limitations

Social

Toots

@fuzion@types.pl

@FuzionLang

Tutorial

➡ Modern Syntax

Classic syntax using { }

▶ Playground

How to...

Install Fuzion
Use Fuzion
Get editor support

Sample Code

Recent commits

Reference

Talks & Paper