# Fanx编程语言

2021-4

chunquedong

# 历史

## Fantom语言

- 2005年Brian Frank和Andy Frank开始写Fan，为了Java和.NET能共享代码
- 2006年编译器实现自举，发布1.0
- 2008年开始支持Javascript编译目标
- 2009年改名为Fantom

## Fanx语言

- 2010年开发slan和vase开发框架
- 2018年改进Fantom诞生Fanx

# 对Fantom的修改

## 语法

- 泛型
- 扩展方法
- Async/await
- Local return
- Checked动态调用
- Runtime Immutable
- Closure推断
- 命名参数
- 中文编程支持
- ……

https://github.com/fanx-dev/fanx/blob/master/doc/DiffFantom.md

## 运行时

- 重写标准库，使用Fantom语言
- 新的构建脚本
- 编译到C源码
- 解释虚拟机

# 熟悉的语法

没有为了不同而不同

```
class Main {
    static Void main() {
        echo("Hello World Fanx")
    }
}
```

# 混合静态和动态类型

动态调用

```
//static invoke
p.age = 10
p.say("A")

//dynamic invoke
p->age = 10
p->say("A")
```

类型推断

```
Int a = 1
a := 1
```

# Null安全

No more NullPointerException
Fantom invent it in 2006

```
Str a = ""     // never stores null
Str? b = null // might store null
```

# 强不可变性

```
const Str p         //immutable
const StrBuf p      //compiler error
readonly StrBuf p //shallow immutable
```

# 闭包/lambda

```
files = files.sort |a, b| { a.modified <=> b.modified }
```

# 安全的并发

- Actor并发模型
- No data-race
- No deadlock

```
actor := Actor |msg| { echo(msg) }
actor.send("Hi")
```

# 声明式编程

- 描述做什么，而不是怎么做
- 序列化格式是代码语法的子集

```
Window
{
  title = "Demo"
  size = Size(600,600)
  GridPane
  {
    numCols = 2
    Label { text="label1" },
    EdgePane
    {
      top    = Button { text = "top" }
      left   = Button { text = "left" }
      right  = Button { text = "right" }
      bottom = Button { text = "bottom" }
      center = Button { text = "center" }
    }
  },
}
```

# 泛型

- 类型擦除
- 协变逆变free

```
class Foo<T> {
  T? t
  T get() { return t }
}

foo := Foo<Str>()
foo.t = "abc"
```

# Async/Await协程

人类友好的回调

```
async Void foo(Int id) {
  user := await getUser(id)
  image := await getImage(user.image)
  imageView.image = image
}
```

```
Void foo() {
    a := requestA()
    b := requestB()
    c := requestC(await a, await b)
    view.text = c
}
```

```
void foo(int id) {
    requestUserInfo(id, (user)->{
        if (err) ...
        requesetImage(user.image, (image)->{
            if (err) ...
            mainThread.post(()-> {
                imageView.image = image
            })
        });
    });
}
```

```
Void foo() {
    lock
    resultA
    resultB

    requestA |a| {
        lock {
            resultA = a
            if (resultB != null) {
                requestC(resultA, requestB) |c| {
                    mainThread.post || {
                        view.text = c
                    }
                }
            }
        }
    }

    requestB |b| {
        lock {
            resultB = b
            if (resultA != null) {
                requestC(resultA, requestB) |c| {
                    mainThread.post || {
                        view.text = c
                    }
                }
            }
        }
    }
}
```

# 模块化

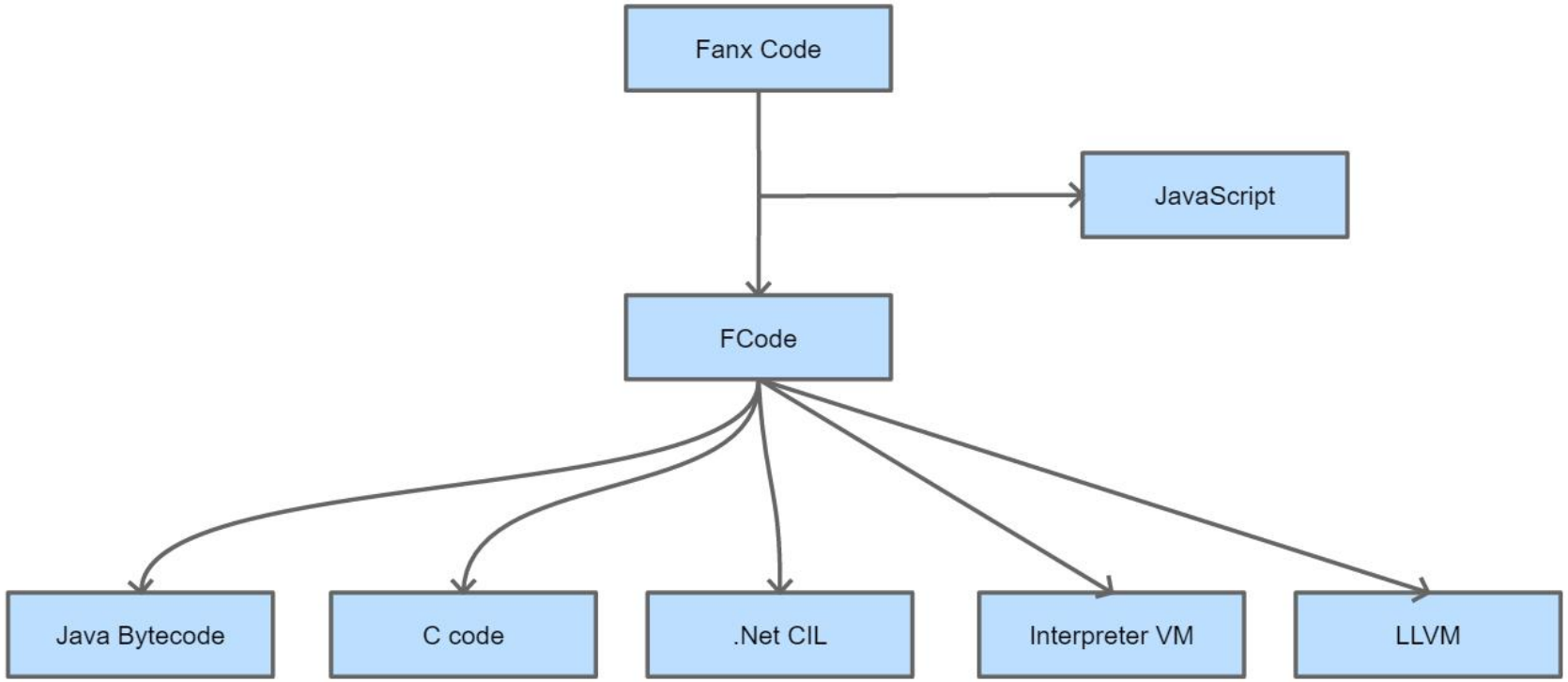
内建模块化支持

- 把程序模块、命名空间、部署单元、版本化单元统一在一起，称为*pod*。
- 软件由这些*pod*通过明确的依赖关系组织在一起
- 包管理工具和*Pod*仓库http://eggbox.fantomfactory.org/

构建脚本

```
podName = testlib
summary = test lib
version = 2.0
srcDirs = test/,fan/
depends = sys 1.0, std 1.0, reflect 1.0
```

# 多编译目标




.net 后端不再维护，LLVM后端WIP

# 优雅的API

Java最大的缺陷不是语言，而是API。

```
//Java
BufferedReader br = new BufferedReader(new FileReader("file.txt"));
try {
    StringBuilder sb = new StringBuilder();
    String line = br.readLine();
    while (line != null) {
        sb.append(line);
        sb.append(System.lineSeparator());
        line = br.readLine();
    }
    String everything = sb.toString();
} finally {
    br.close();
}

//Fanx
File(`file.txt`).readAllStr
```

# Fanx标准库

## Fanx核心库

包括编译器、标准库、并发库。除了核心类库以外还包括：

- 常用的容器
- Actor并发模型
- IO库
- 日期时间
- 日志
- 单元测试
- 正则表达式
- JSON
- CSV格式解析
- 序列化
- 压缩
- URL解析
- Base64/MD5
- ...

## Fontom库

由Fantom团队维护的库，在Fanx上测试过的有：

- 网络库
- Email库
- **Wisp http服务器**
- XML解析
- 包管理
- 语法高亮
- API文档生成
- DomKit
- SQL数据库接口

# 应用开发框架

## Slan

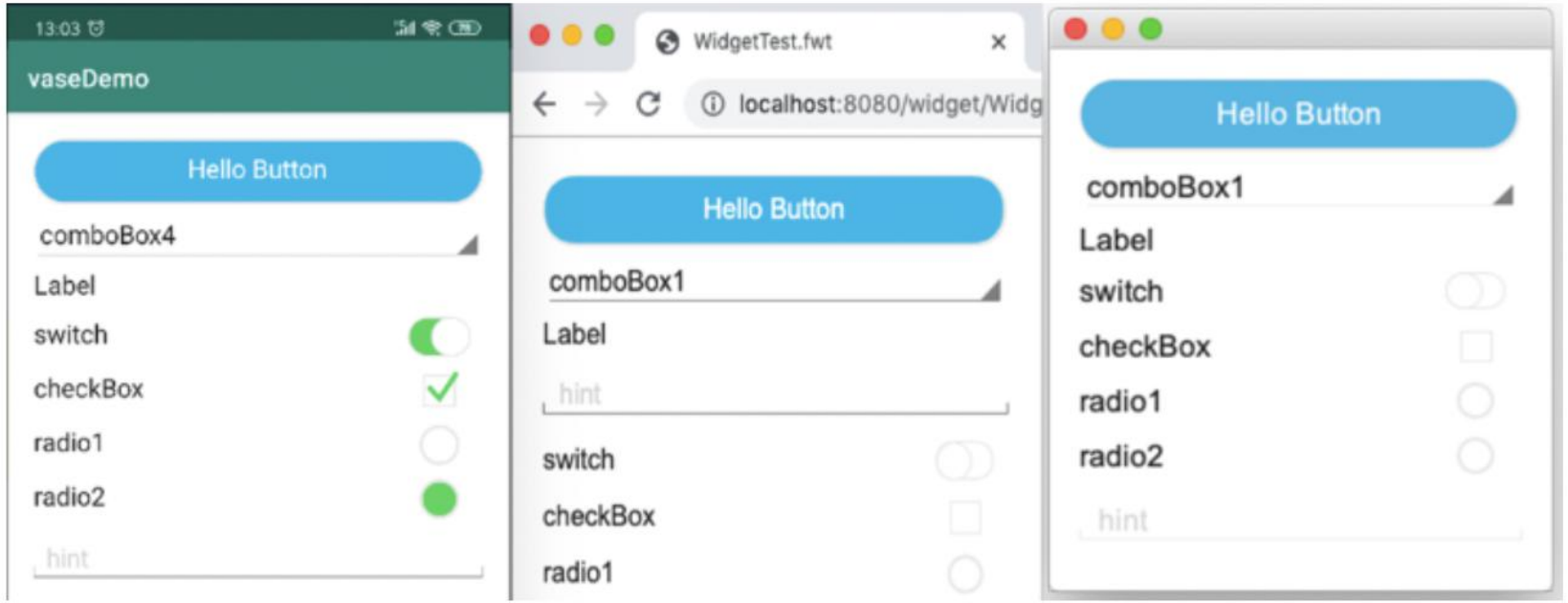
- slan是web后端开发框架。包括关系数据库接口、ORM对象关系映射工具、URL路由框架、FSP模版引擎、Restful和MVC、代码热加载、验证码生成等。

## Domkit

- domkit是基于HTML5的UI开发框架，用来替代fwt（基于SWT的UI库）。
- domkit使用标准的Web技术构建应用。

# Vase跨平台App开发框架

- 移动端优先，原生跨平台UI开发框架。支持开发Android/iOS/Web浏览器/桌面APP开发。
- 使用vase 3D图形库，可以同时开发OpenGL和WebGL应用。
- 基于async/await的Http请求库，同样支持所有平台。



# 网络和数据库引擎(实验性)

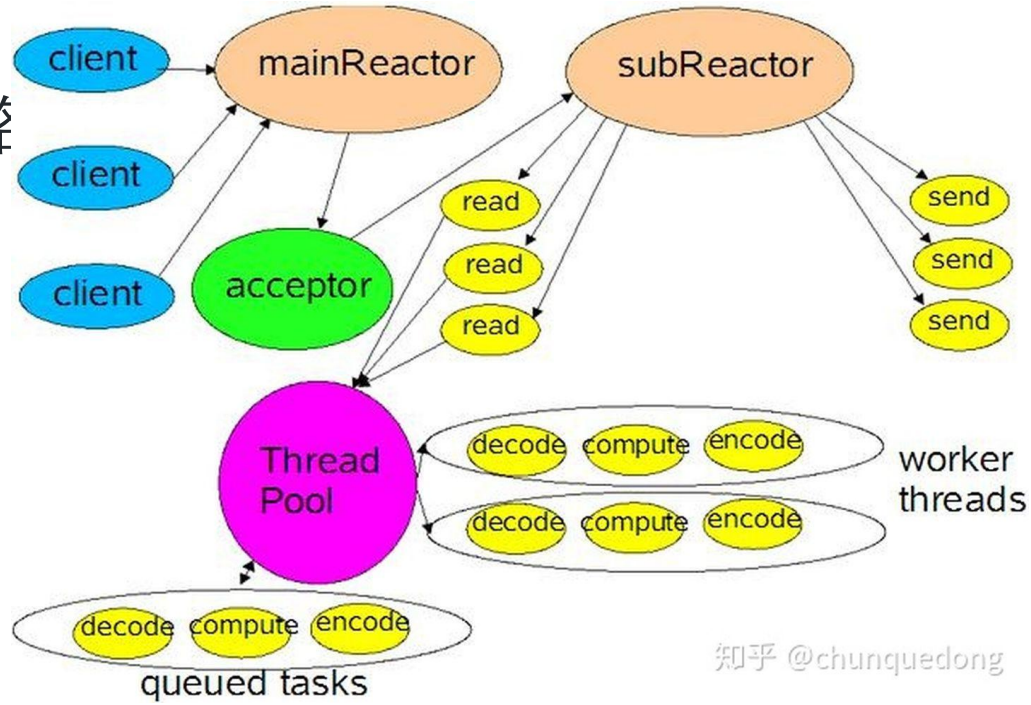
## asyncServer

async/await coroutine + Java NIO网络

## axdb2

高性能key-value数据库

- Raft分布式一致性
- 存储引擎LSM tree and B+tree
- async/await非阻塞IO网络框架



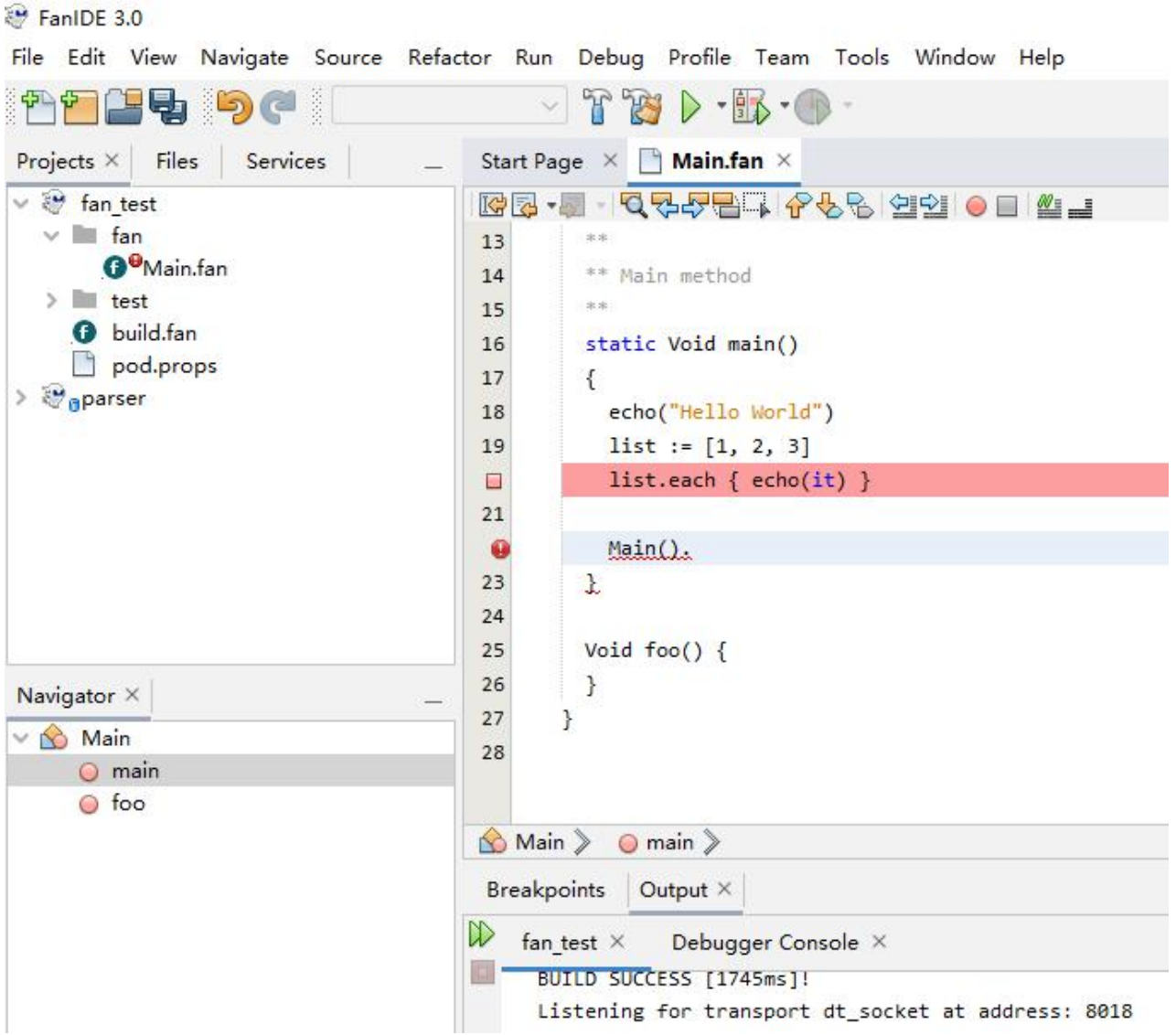
# 开发工具

## FanIDE

Fanx集成开发环境，基于Netbeans 。

## 其他IDE

- F4 （eclipse based）
- Visual Studio Code
- Sublime Text 3
- ...



# 代码实现

## Compiler

- Compiler: 编译器核心
- CompilerJava: java FFI插件
- CompilerJS：转化为JS插件
- Build：构建工具

## Parser

- 用于IDE语法解析，将来会替代Compiler

## JavaEmit

- 动态生成java bytecode

## funRun

- gen：Compile to C
- vm：The interpreter VM
- Llvm：LLVM compiler

# Fanx/Fantom状态

- Production ready on JVM backend
- 2008用于商业软件开发

- 1.5k个单元测试，2000k case验证
- 5.5k git commit

- Academic Free License 完全开源
- 无第三方依赖

# 将来的工作

- iOS运行时
- 改进IDE
- .fanx(break change)

# Q&A

http://fanx.info
https://fantom-lang.org/
https://fantom.org/