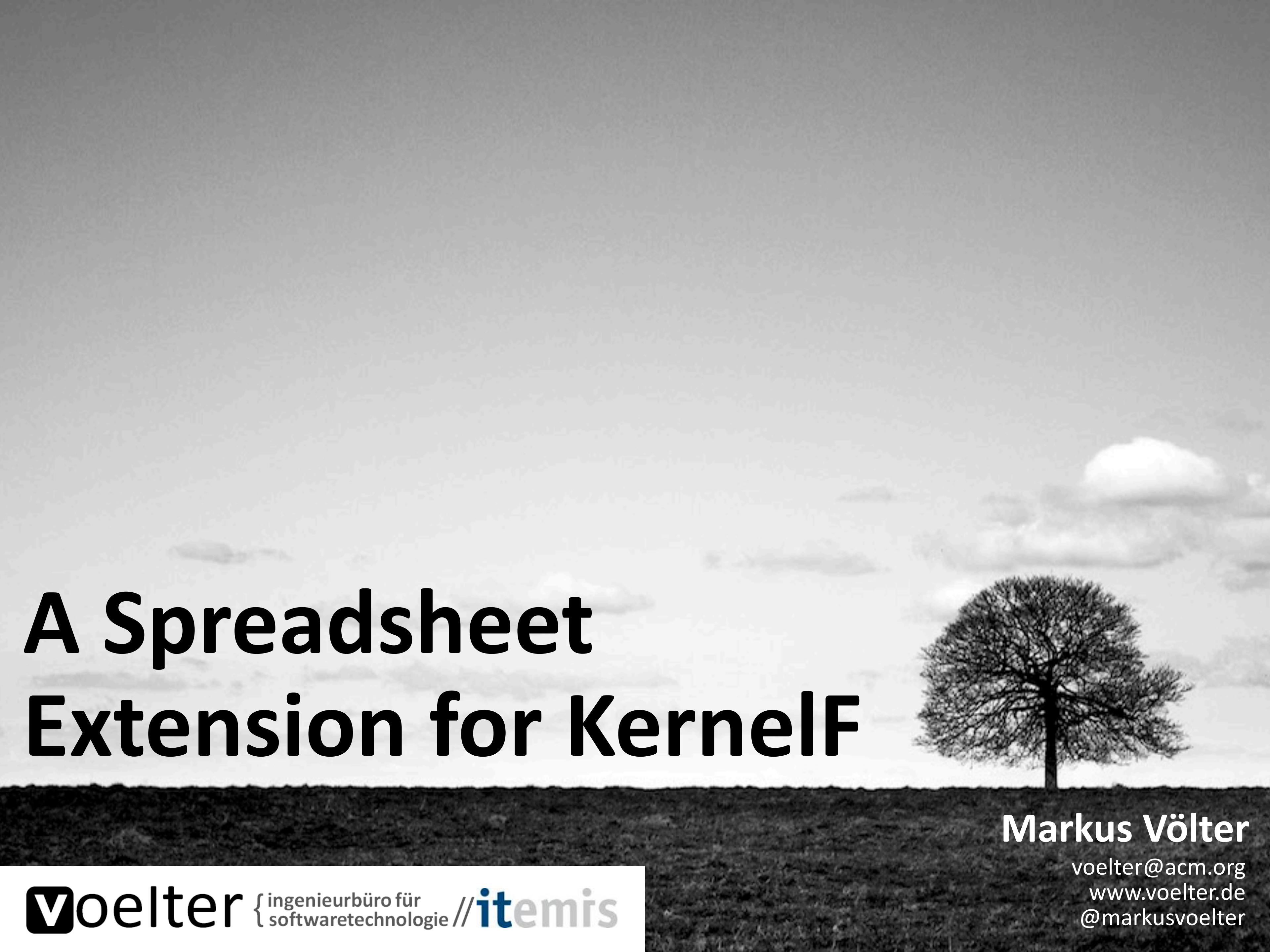


A Spreadsheet Extension for KernelF

Markus Völter
voelter@acm.org
www.voelter.de
@markusvoelter

voelter {ingenieurbüro für softwaretechnologie // itemis

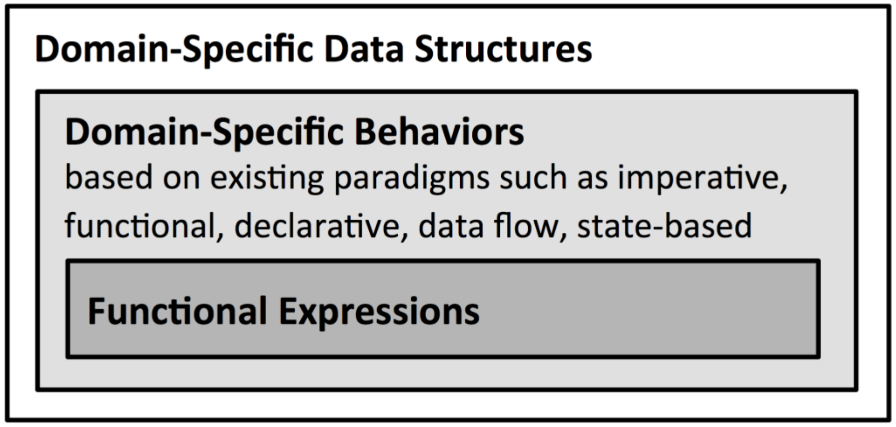
1

KernelF

kernelf

functional • embeddable • extensible

A functional language to be used at the core of DSLs based on MPS.


Domain-Specific Data Structures
Domain-Specific Behaviors
based on existing paradigms such as imperative, functional, declarative, data flow, state-based
Functional Expressions

Expressions!

Primitive Types
Numbers, Booleans, Strings, Enums, Records

Arithmetic Operations
+ - * /

Comparison Operations
> >= < <= == !=

Functions
Definitions, Calls, Higher-Order

KernelF is a functional base language

Core Features

The usual types and literals

User-defined types: tuples, records and enums

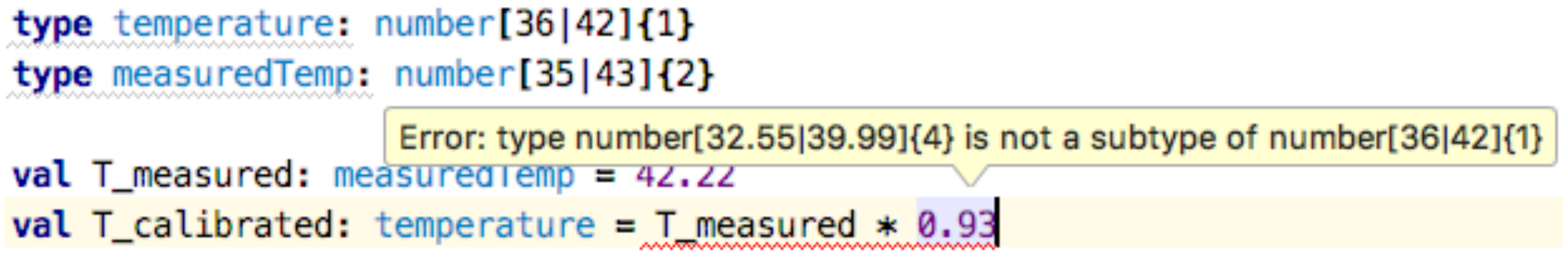
Option Types

Attempt Types

Higher-Order Functions and Lambdas

Effects Tracking

Number Types with Ranges



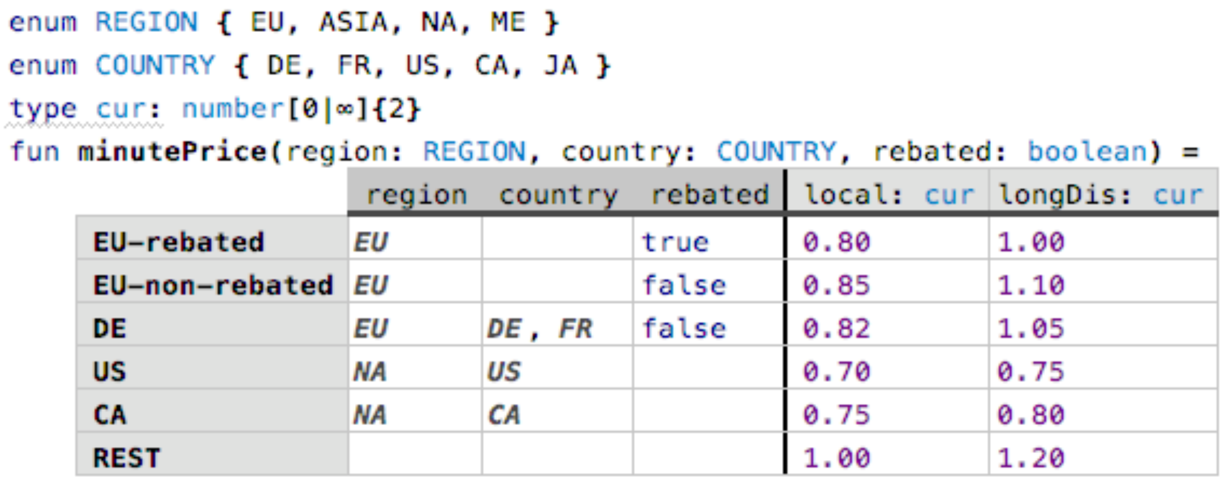
```
type temperature: number[36|42]{1}
type measuredTemp: number[35|43]{2}

val T_measured: measuredTemp = 42.22
val T_calibrated: temperature = T_measured * 0.93
```

Error: type number[32.55|39.99]{4} is not a subtype of number[36|42]{1}

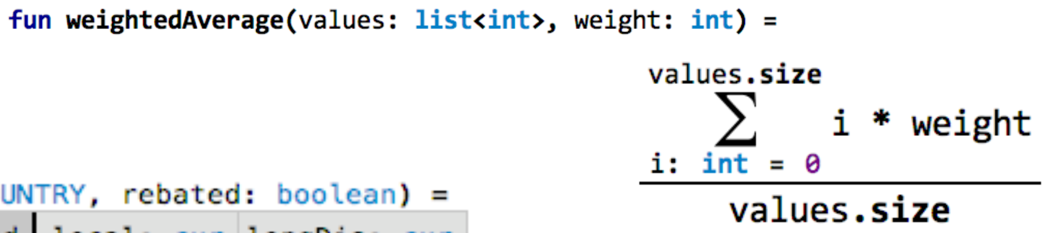
Extended Features

Type Tags and Units
Several Forms of Decision Tables
Math Notation
Natural Language Function Calls



```
enum REGION { EU, ASIA, NA, ME }
enum COUNTRY { DE, FR, US, CA, JA }
type cur: number[0|∞]{2}
fun minutePrice(region: REGION, country: COUNTRY, rebated: boolean) =
```

	region	country	rebated	local: cur	longDis: cur
EU-rebated	*EU*		true	0.80	1.00
EU-non-rebated	*EU*		false	0.85	1.10
DE	*EU*	*DE , FR*	false	0.82	1.05
US	*NA*	*US*		0.70	0.75
CA	*NA*	*CA*		0.75	0.80
REST				1.00	1.20

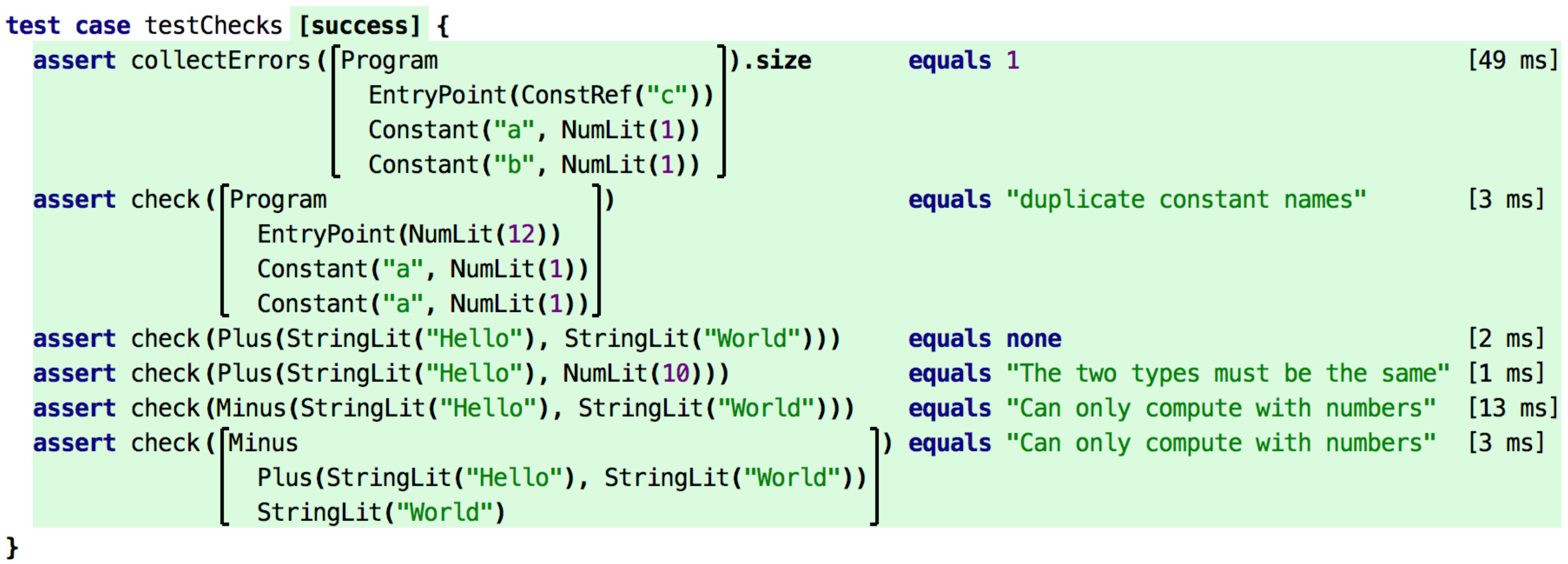


```
fun weightedAverage(values: list<int>, weight: int) =
```

$$\frac{\sum_{i:\ \mathrm{int}\ =\ 0}^{\mathrm{values.size}} i * \mathrm{weight}}{\mathrm{values.size}}$$

Integrated Interpreter

Used for testing and simulation



```
test case testChecks [success] {
  assert collectErrors([Program                      ]).size      equals 1                                  [49 ms]
                       [  EntryPoint(ConstRef("c"))]
                       [  Constant("a", NumLit(1)) ]
                       [  Constant("b", NumLit(1)) ]
  assert check([Program                  ])                       equals "duplicate constant names"         [3 ms]
               [  EntryPoint(NumLit(12)) ]
               [  Constant("a", NumLit(1))]
               [  Constant("a", NumLit(1))]
  assert check(Plus(StringLit("Hello"), StringLit("World")))      equals none                               [2 ms]
  assert check(Plus(StringLit("Hello"), NumLit(10)))              equals "The two types must be the same"   [1 ms]
  assert check(Minus(StringLit("Hello"), StringLit("World")))     equals "Can only compute with numbers"    [13 ms]
  assert check([Minus                                        ])   equals "Can only compute with numbers"    [3 ms]
               [  Plus(StringLit("Hello"), StringLit("World"))]
               [  StringLit("World")                          ]
}
```

2

Spreadsheets

Spreadsheets Characteristics

Functional / Computation-as-State.
Good notation for many use cases.
Everybody knows it.

Spreadsheet as **Database**	Spreadsheet as **Calculator**	Spreadsheet as **Language**

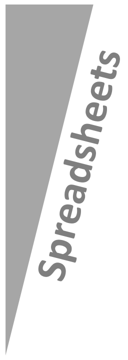
No notion of instantiation.
Limited typing/schema support.
A single notation for everything.
No growability towards domains.

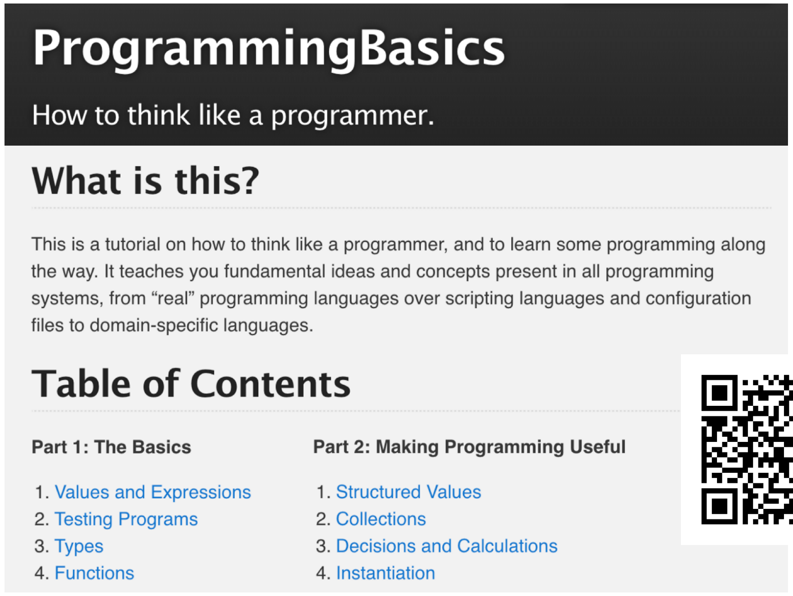
$\Bigg\}$ Scale with Size and Complexity

Why Spreadsheets in KernelF

Teaching

Values and Expressions
Testing Programs
Types
Functions
Structured Values
Collections
Decisions and Calculations
Instantiation





ProgrammingBasics

How to think like a programmer.

What is this?

This is a tutorial on how to think like a programmer, and to learn some programming along the way. It teaches you fundamental ideas and concepts present in all programming systems, from “real” programming languages over scripting languages and configuration files to domain-specific languages.

Table of Contents

https://markusvoelter.github.io/ProgrammingBasics/

As part of DSLs

A kind of tabular REPL

Instantiation.
Schema support.
Just one notation.
Extensible.

DEMO

http://127.0.0.1:63320/node?ref=r%3Ae79a89ea-18bd-43a3-a4a6-fdfa93b98a51%28playground.wgld%29%2F863326562414371328

Open Issues

More scalable table editor implementation.

Sascha is on it :-)

Reactive Interpreter.

Waiting for (customer) funding.

Cleanup.

Will be done soon.