Programming Language Oenology for Trolls

marv

2018-06-15

Expressions

```
1
1 + 2
1 + 2 * 3
```

Expressions

```
1
1 + 2
1 + 2 * 3

f(x)
a.b
a.f(x)
```

Expressions

```
1
1 + 2
1 + 2 * 3

f(x)
a.b
a.f(x)

1 plus 2
1.plus(2)
add(1, 2)
add 1 2
1 2 +
```

Expressions

```
1
1 + 2
1 + 2 * 3

f(x)
a.b
a.f(x)

1 plus 2
1.plus(2)
add(1, 2)
add 1 2
1 2 +

x = 1
y = a.f(x)
f = a.f
z = f(x)
```

Control flow

```
if (!x.is_empty()) {

while (true) {

for (;;) {
```

Control flow

```
if (!x.is_empty()) {

while (true) {

for (;;) {

unless (x.is_empty()) {

loop {
```

Functions

Currying

```
(x, y) => ...
x => y => ...
```

Functions

Currying

```
(x, y) => ...
x => y => ...
```

Dispatch

```
a.f(b)
f(a, b)
element.accept(visitor)
visitor.visitElement(element)
```

Typing

Typing

```
1
1 / 2
```

Typing

```
1
1 / 2

if (x == 1) true else ""
x == 1 ? true : ""
```

Typing

```
1
1 / 2

if (x == 1) true else ""
x == 1 ? true : ""

1 - "2"
1 / 2.0
```

Polymorphism

```
def f(x: int)
def f(x: float)
def f()
def f(x: Int|Float|Null = null)
```

Polymorphism

```
def f(x: int)
def f(x: float)
def f()
def f(x: Int|Float|Null = null)
```

```
Map<String, Integer>
map[string]int
```

Polymorphism

```
def f(x: int)
def f(x: float)
def f()
def f(x: Int|Float|Null = null)

Map<String, Integer>
map[string]int

S <: T
Integer <: Rational <: Real
... <: Object
Null <: ...
{x: Int; y: Int} <: {x: Int}
```

Variance

```
String[] x;
Object[] y = x;
```

Variance

```
String[] x;
Object[] y = x;

def generate(): Rational
override def generate(): Integer
def consume(x: Rational)
override def consume(x: Real)
```

End