Сошников Дмитрий Валерьевич
к.ф.-м.н., доцент
dmitryso@microsoft.com

Функциональное программирование

Факультет инноваций и высоких технологий
Московский физико-технический институт

Лекция 1

Определение и краткая история функционального программирования

Обо мне

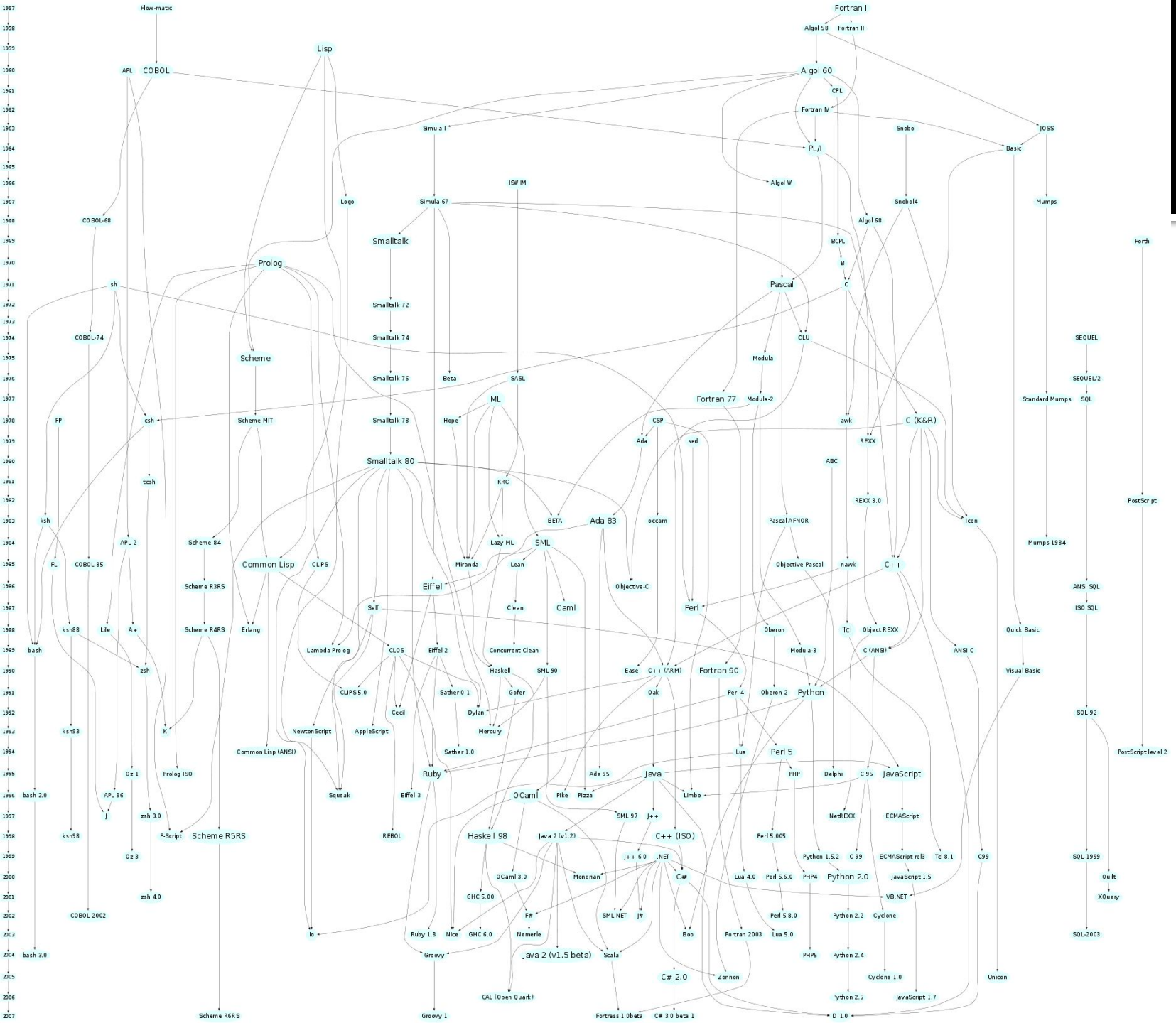
- Майкрософт Россия, академический евангелист
- Кандидат физ.-мат. наук
  - Распределенные интеллектуальные системы с явным представлением знаний
  - Интеллектуальная реструктуризация социальных сетей на основе онтологий
  - Семантически-ориентированые системы (Semantic Wiki)
- Кафедра Вычислительной математики и программирования МАИ (доцент)
  - Логическое программирование
  - Искусственный интеллект
  - Студенческая лаборатория MAILabs (www.mailabs.ru)
- ФИВТ

http://blogs.gotdotnet.ru/personal/sos

Какие языки программирования вы знаете?

- Assembler (x86, …)
- C, C++, C#, Java
- Pascal
- …

- Brainfuck?
- FORTH?
- LISP, FP, ML, Haskell, OCaml, F#, …



Немного истории

Функциональное программирование

- Парадигма программирования, которая рассматривает выполнение программы как вычисление математических функций (выражений)
  - Неизменяемые данные, нет состояния среды
- Стиль программирования, позволяющий писать программы, свободные от ошибок
- Язык программирования F# (и целое семейство «странных» языков вместе с ним: ML, Haskell, …)

«Классическое программирование»

- Императивное – мы говорим компьютеру, как решать задачу (что делать)
- Основной акцент – манипулирование ячейками памяти
  - Оператор присваивания
- Функции как способ декомпозиции задачи на более простые

Обратимся к истории


```
0000   0A 12 1F 4B C3 E0 EE F1
0008   C3 1D 23 17 F2 00 0C 0D
0010   …
```

```
      MOV   AX, [ARG1]
      ADD   AX, [ARG2]
      MOV   [RES], AX
      JMP   NEXT
ARG1:       DB    10
ARG2:       DB    20
RES: DB     0
NEXT:       …
```

```
            S = 0
            DO 10 I=1,10
            S = S + I*I
10          CONTINUE
```

1954-57 г., Дж.Бэкус

- FORTRAN
- язык ассемблера
- машинные коды
- программирование переключателей

1950 1960 1970 1980 1990 2000 2010

- Первый язык программирования высокого уровня – ФОРТРАН – был создан Дж.Бэкусом, чтобы математики могли программировать на уровне формул.

Светлая сторона силы!

```
def fac = eq 0 →1;
*○[id,fac○(-○ [id,1])]
```


1958 г., Дж.Маккарти

- ♦ LISP
- FORTRAN
- язык ассемблера
- машинные коды
- программирование переключателей

1977 г., Дж.Бэкус

- ♦ FP

```
(defun factorial (n)
   (if (<= n 1) 1
   (* n (factorial (- n 1)))))
```

1950 1960 1970 1980 1990 2000 2010

- Позже Дж.Бэкус пошел дальше и предложил язык FP, в котором формулы более соответствовали математическому понятию функции

Посмотрим пример!

- Вычисление факториала:

```
function fact(x:integer):integer;
var i, r : integer;
begin
  r:=1;
  for i:=1 to x do r:=r*i;
  fact:=r
end;
```

Pascal

```
let rec fact x =
  if x=1 then 1
  else x*fact(x-1);;
```

```
let rec fact = function
    1 -> 1
  | x -> x*fact(x-1);;
```

F#

Что особенного?

- Определение функции похоже на математическое определение факториала
  - Функциональное программирование имеет очень четкую математическую основу
  - Рассуждение о программах: доказательство корректности, …
- Определение последовательности действий – рекурсивно
  - При умелом программировании не ведет к падению эффективности (компилятор сводит к итерации)
- **Отсутствует оператор присваивания**
  - let имеет другую семантику – связывание имен
  - Будучи один раз связанным, имя не может менять свое значение (в рамках области видимости)
  - А это значит – нет побочных эффектов!
  - Раз в императивной программе 90% - это операторы присваивания, то функциональные программы на 90% короче!

Функциональный стиль

```
function fact(x:integer):integer;
begin
  if x=1 then fact:=1
  else fact:=x*fact(x-1)
end;
```

- Это не «чистая» императивная программа.
  - В «чистых» императивных языках (ФОРТРАН) нет рекурсии
- Нет операторов присваивания
  - «:= » -это возврат результата из функции, а не присваивание