X конференция АП КИТ
Преподавание ИТ в России

Биллиг В.А.
Vladimir.Billig@tversu.ru

Так учат программированию в ETH

План

- Так учат в ETH
- Почему Eiffel
- Почувствуй класс
- Курсы на Интуит

ETH

- Eidgenossische Technische Hochschule, Цюрих
- Один из старейших университетов Европы
- Входит в пятерку лучших университетов
- Вирт в ETH

Бертран Мейер

- Бертран в ETH
- язык Eiffel
- Контрактное программирование
- Книги
- Связи с Россией

Язык для обучения программированию
От Паскаля к Эйфелю

- Паскаль – язык структурного программирования
- Эйфель – язык объектного программирования
- Как успеть за современными вызовами
- Король среди языков для обучения программированию – язык Паскаль – уже умер. Новый король еще не назван.

Как учат в ETH

- **Три кита**
  - Объекты
  - Контракты
  - Извне – внутрь (Обращенный учебный план)

Классика обучения
От простых программ к сложным системам

- Переменные, объявления
- Операторы: присваивание, выбор, цикл
- Модульность: процедуры, функции
- Модульность: классы
- Отношения между классами
- Сложные системы
- Проблемы сложных систем

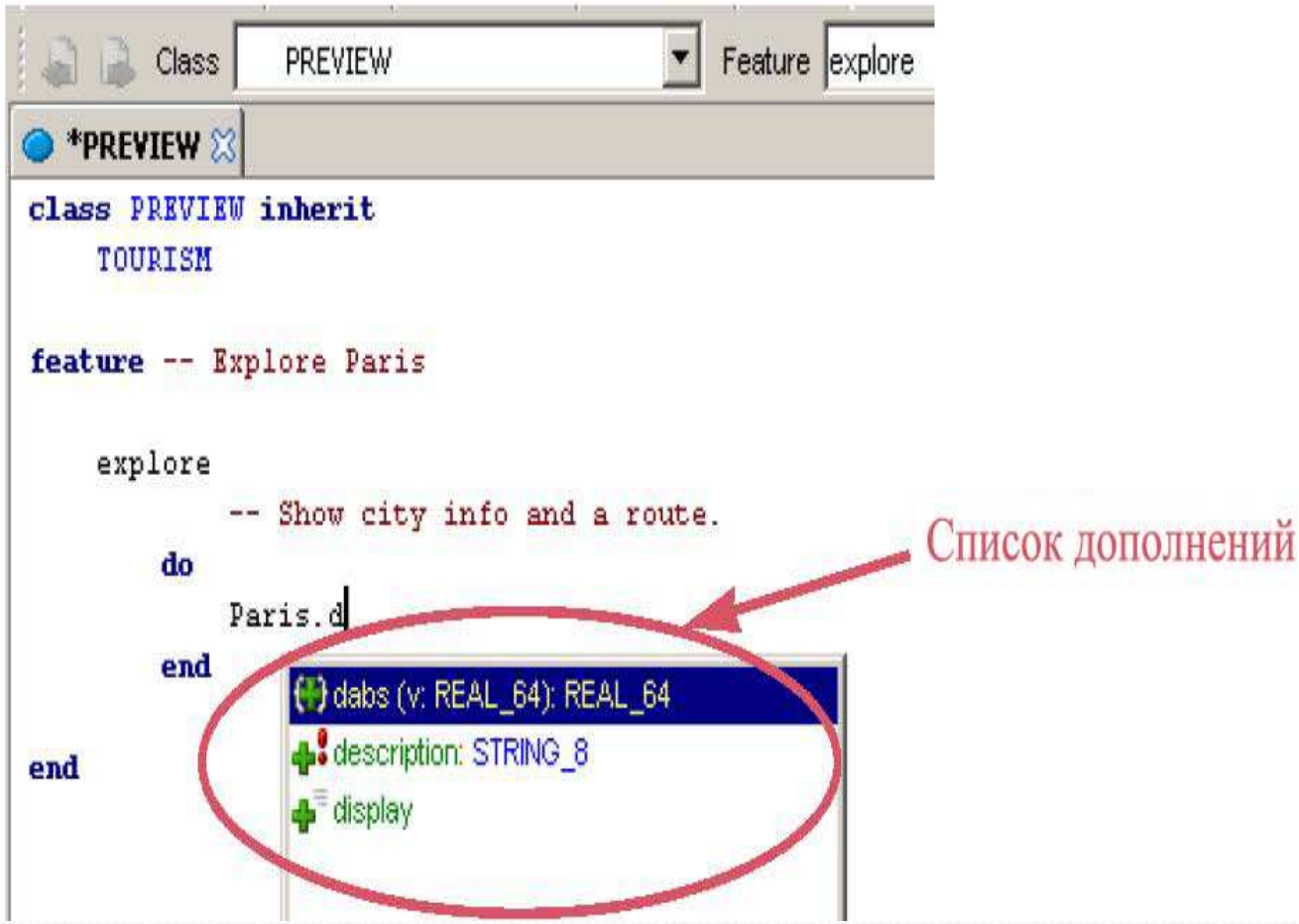
Обучение в ETH
Вначале сложная система

- Система Traffic – система общественного транспорта
- Изучаем интерфейсы
- Контракты как часть интерфейса

Первая программа

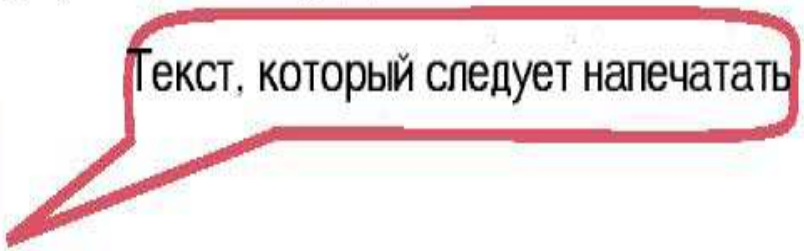
- Отобразить карту Парижа, включая карту метро.
- Подсветить расположение музея Лувр.
- Подсветить одну из линий метро – линию 8.
- Выполнить анимацию подготовленного маршрута.

Первая программа



Первая программа

```
-- Показать некоторую информацию о городе.
do
    Paris.display
    Louvre.spotlight
    Line8.highlight
    Route1.animate
end
```



Вызов компонента

- Объект – единица данных
- Компонент (feature) – операция, применимая к данным
- Вызов компонента x.f – фундаментальная конструкция ООП (x – объект, f – компонент)
- Paris.display (Paris - объект, display – компонент)

Запросы, команды, интерфейс класса

- Класс Line (пример запроса)

i_th (*i: INTEGER*): *STATION*

-- *i*-я станция на этой линии

require

not_too_small: i >= 1

not_too_big: i <= count

Запросы, команды, интерфейс класса

- Пример команды класса Line

remove_all_segments

-- Удалить все станции кроме южного конца.

ensure

only_one_left: *count* = 1

both_ends_same: *south_end* = *north_end*

Управляющие структуры

- Управляющая структура выбора

  if условие then

  составной оператор 1

  else

  составной оператор 2

  end
- Вариации if
- Inspect … when - множественный выбор

Управляющие структуры

- Структура цикла

from

<инициализация >

invariant

<инварианты цикла>

until

<условие завершения цикла>

loop

<тело цикла>

variant

<вариант цикла>

end

Управляющие структуры и структуры данных. Пример

```
current_station STATION
from
        Line8.start
invariant
        (not is_empty) implies (not is_before)
        -- точка отображена для всех станций перед позицией курсора
until
        Line8.is_after
loop
        current_station := Line8.item
        if current_station.is_exchange  then
                show_blinking.spot(current_station.location)
        else
                show_spot(current_station.location)
        end
        Line8.forth
variant
        Line8.count – Line8.index + 1
end
```

Отвечая на вызовы

- Погружение в сложную систему.
- Ранее знакомство с центральными понятиями ООП - объектами, запросами, командами.
- Ранее знакомство с абстракцией – интерфейсами методов, контрактами.
- Обучение на хороших примерах.
- Поддержание интереса студентов с разным уровнем подготовки.
- Первый шаг к инженерии программ начинается с первой программы.

Почему Эйфель?

- Создавался как язык, в полной мере отвечающий концепциям ООП, - центральной технологии создания современных программных систем
- Отвечает цели обучения – создание систем:
- Высокого качества;
- Повторно используемых;
- Живущих долго;
- Меняющихся в ответ на требования времени;
- Корректность которых поддерживается с самого начала.

Почему Эйфель

- *Язык, в котором концепции ООП выражены на языке программирования*

12 отличий

- Контракты;
- Множественное наследование;
- Агенты;
- Правила доступа;
- Обработка исключений;
- Статический контроль null – вызовов;
- Ссылочные и значимые типы;
- Две роли класса;
- Унифицированный доступ;
- Перегрузка;
- Организация цикла;
- Организация множественного выбора.

Роль контрактов

- Предусловия и постусловия методов
- Инварианты класса
- Инварианты и варианты цикла
- Контракты и наследование
- Контракты и обработка исключительных ситуаций

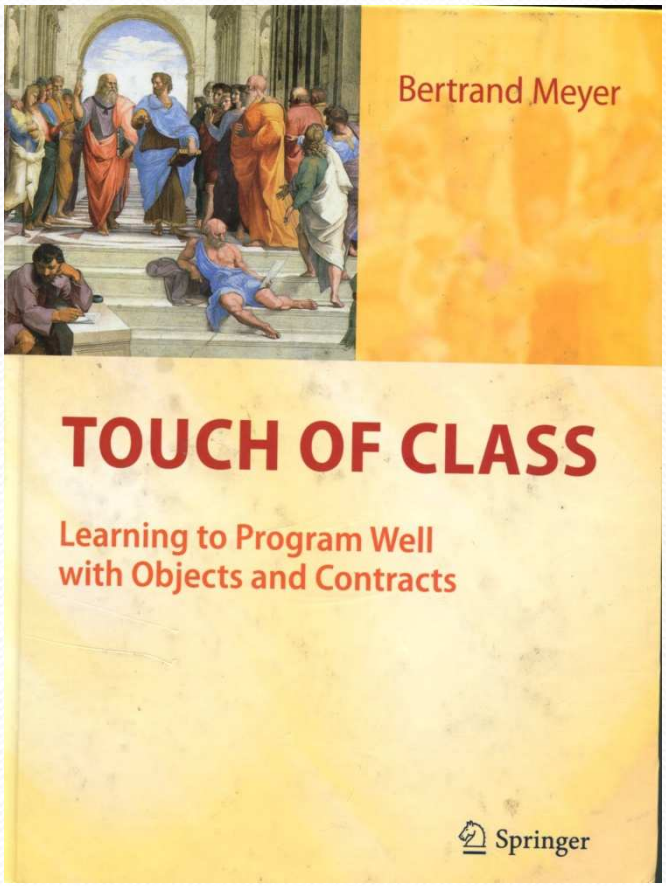
Параллелизм

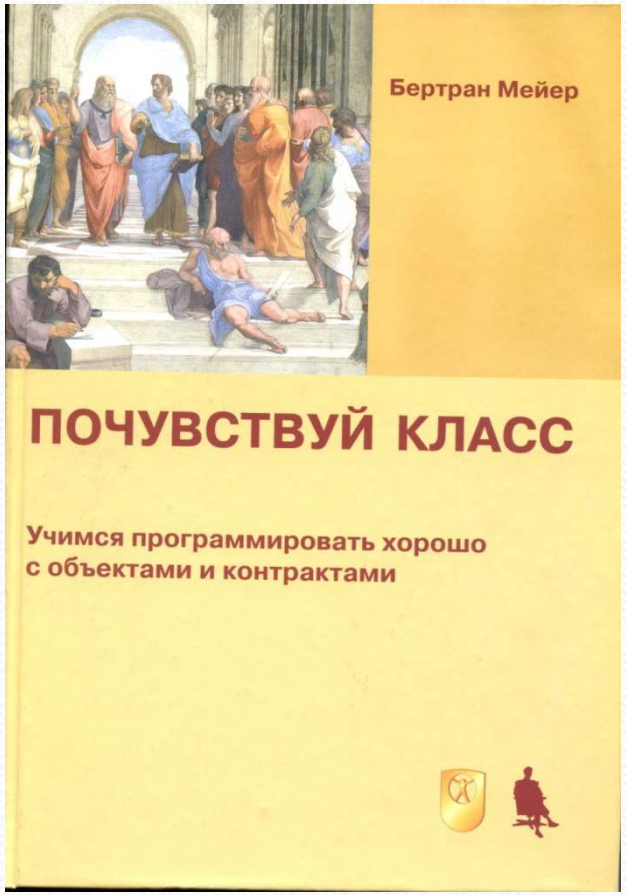
- Модель SCOOP (Simple Concurrent Object-Oriented Programming)
- Активная теоретическая работа по развитию модели SCOOP продолжается

Развитие языка

- Эйфель живой, активно развивающийся язык

«Touch of Class»
«Почувствуй класс»





Что есть программирование?

- Искусство
- (Кнут – «Искусство программирования)
- Наука
- (Грис – «Наука программирования»)
- Инженерия
- (Соммервилл – «Инженерия программного обеспечения»
- **Искусство, Наука, Инженерия**

«Почувствуй класс» и программирование

- **Наука программирования**
  - Неразрешимость проблемы остановки
  - Агенты и основы лямбда исчисления
  - Статический контроль и Null ссылки
  - Модель SCOOP
- **Искусство программирования**
  - динамические структуры (списки, деревья, хэш-таблицы)
  - Топологическая сортировка
- **Инженерия программирования**
  - Часть 5: Цель – инженерия программ
  - Доклад Бертрана Мейера на конференции:

Обширный подход к преподаванию программной инженерии

Содержание

- **Часть 1 Основы**
  - Объекты и вызовы, Интерфейс класса, Создание объектов, структуры управления, подпрограммы, Присваивание
- **Часть 2 Как все устроено**
  - Аппаратура, языки, инструментарий
- **Часть 3 Алгоритмы и структуры данных**
  - Сложность алгоритмов, Массивы, Кортежи, Списки, Стеки, Очереди
  - Хеш-таблицы, Рекурсия, Деревья, Топологическая сортировка
- **Часть 4 ОО приемы программирования**
  - Наследование, Полиморфизм, Множественное наследование
  - Агенты, Применение агентов, Лямбда-исчисление, Карринг, Лямбда-исчисление и агенты
  - События, Паттерны
- **Часть 5 Цель – инженерия программ**
- **Часть 6 Приложения**
  - C, C++, Java, C#

Курсы в Интуите

- Основы программирования
  - http://www.intuit.ru/department/se/mbasepr/
- Инструменты, алгоритмы и структуры данных
  - http://www.intuit.ru/department/se/ialgdate/
- Объектно-ориентированное программирование и программная инженерия
  - http://www.intuit.ru/department/se/ooppe/

Интервью с Бертраном Мейером

http://intuit.ru/department/se/mbasepr/0/#clip.1

Итоги
Триада Хоара:**Pred {Prog} Post**

- Prog – это программа обучения студентов, заданная учебником «Почувствуй класс»;
- Post – постусловие, истинность которого означает, что по завершении учебного курса студенты умеют программировать хорошо с объектами и контрактами.
- Pred – предусловие: существует коллектив преподавателей, способных донести эту программу & существует коллектив студентов, способных воспринимать эту программу.

Итоги
Триада Хоара:**Pred {Prog} Post**

- *Опыт ETH доказывает истинность триады*

Вопросы

?