

# Programming Environments (User Interface for Developers)

学部生講義「ユーザインタフェース」

担当: 加藤 淳 / 五十嵐研究室 博士課程3年

<http://junkato.jp/>

@arcatdmz #devenv

# User Interface for ?

- ありとあらゆる人にユーザインタフェースが必要
- “Programmers are people, too.” – Ken Arnold, 20005
  - 「プログラマだって人間だ」
- そう、ユーザインタフェースを作っているプログラマも例外ではありません



**Programming Environments**  
(User Interface for Developers)

# 今回の内容

- プログラミングって何？

- 文字ベースの統合開発環境

Dartmouth BASIC, Visual Studio, Elipse, Code Bubbles, Active Code Completion

- 視覚表現を活用した統合開発環境

Whyline, Barista, Sikuli, Picode, DejaVu

- 特別な目的に最適化された統合開発環境

PureData, Processing, Arduino, d.tools, Gestalt

- 統合開発環境とインターネット

Blueprint, HelpMeOut, Collabode, TouchDevelop, MOOCs

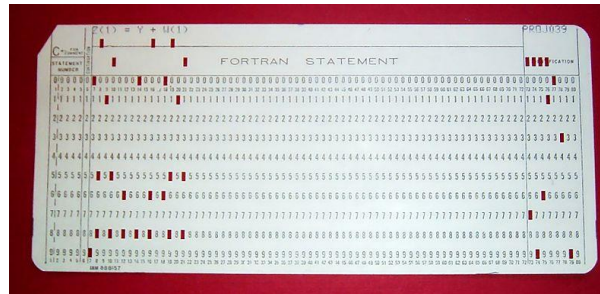
# プログラミングって何？ (1/1)

- コンピュータに、やってほしい手続きを伝える方法
- かつてのプログラミング [～1960年代前半]



記録する内容は  
自力で計算

機械語をカードに記録



実行



プログラミングは全くインタラクティブじゃなかった😓

# プログラミングって何？ (2/2)

- その後のプログラミング [1960年代後半～]

```
public class HelloWorld {  
    // Hello World!  
    public static void  
        main(String[] args) {  
        System.out.println("Hello World!");  
    }  
}
```

スクリーン上で  
高級言語を記述

機械語を電子的に保存

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F	0123456789ABCDEF
000000	CA	FE	BA	BE	00	00	00	33	00	14	07	00	02	01	00	0A	.....3.....
000010	48	85	6C	6C	6F	57	6F	72	6C	64	07	00	04	01	00	10	HelloWorld.....
000020	6A	61	76	61	2F	6C	61	6E	67	2F	4F	62	6A	65	63	74	java/lang/Object
000030	01	00	06	3C	69	6E	69	74	3E	01	00	03	28	29	56	01	...<init>...(V.
000040	00	04	43	6F	64	65	0A	00	03	00	09	0C	00	05	00	06	..Code.....
000050	01	00	0F	4C	69	6E	65	4E	75	6D	62	65	72	54	61	62	...LineNumberTab
000060	6C	65	01	00	12	4C	6F	63	61	6C	56	61	72	69	61	62	le...LocalVariab
000070	6C	65	54	61	62	6C	65	01	00	04	74	68	69	73	01	00	leTable...this..
000080	0C	4C	48	65	6C	6C	6F	57	6F	72	6C	64	3B	01	00	04	..LHelloWorld;..
000090	6D	61	69	6E	01	00	16	28	5B	4C	6A	61	76	61	2F	6C	main...([Ljava

実行

Console

<terminated> HelloWorld  
Hello World!

エディタ

コンパイラ

デバッガ

インタラクティブにプログラムを作れるようになった😊

# 文字ベースの 統合開発環境

General text-based IDE

学部生講義「ユーザインタフェース」

「Programming Environments (User Interface for Developers)」

# 視覚表現を活用した 統合開発環境

- **Dartmouth BASIC:** 史上初の統合開発環境
- **Visual Studio, Eclipse, ...**
- **Code Bubbles:** コードナビゲーションを直感的に
- **Active Code Completion:** コード補完を使いやすく

# 文字ベースの統合開発環境 (1/2)

Dartmouth Time-Sharing System (DTSS) [1964-]

- ソースコードの読み込み・保存、編集、コンパイル、実行が一通りできた初めての統合開発環境

```
GW-BASIC 3.23
(C) Copyright Microsoft 1983,1984,1985,1986,1987,1988
60300 Bytes free
Ok
_

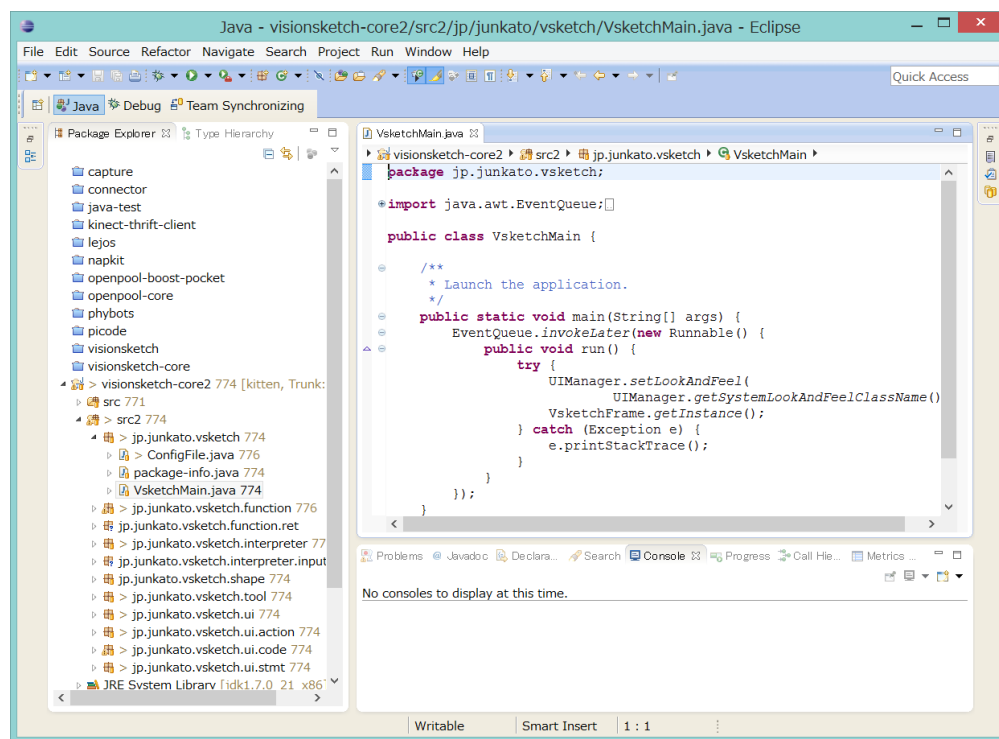
1LIST 2RUN← 3LOAD" 4SAVE" 5CONT← 6,"LPT1 7TRON← 8TROFF← 9KEY 0SCREEN
```

GW-BASIC  
DTSSに似た  
Microsoft製方言



# 文字ベースの統合開発環境 (2/2)

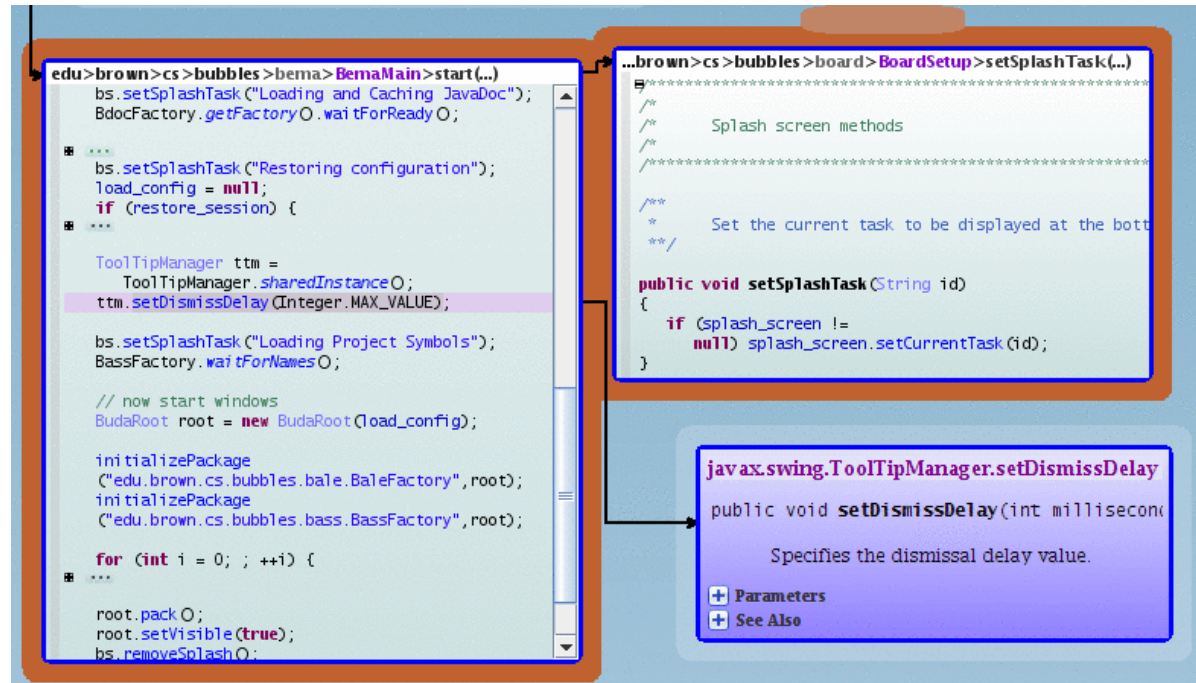
Visual Studio (Microsoft C) [1983-], Eclipse (IBM VisualAge) [1984-], Xcode [2003-], ...



- だいたいどれも同じような機能を備えている
- 文字列の高級言語を編集できるエディタ
- メモリ内容を文字列で表示できるデバッガ
- ...

# Code Bubbles

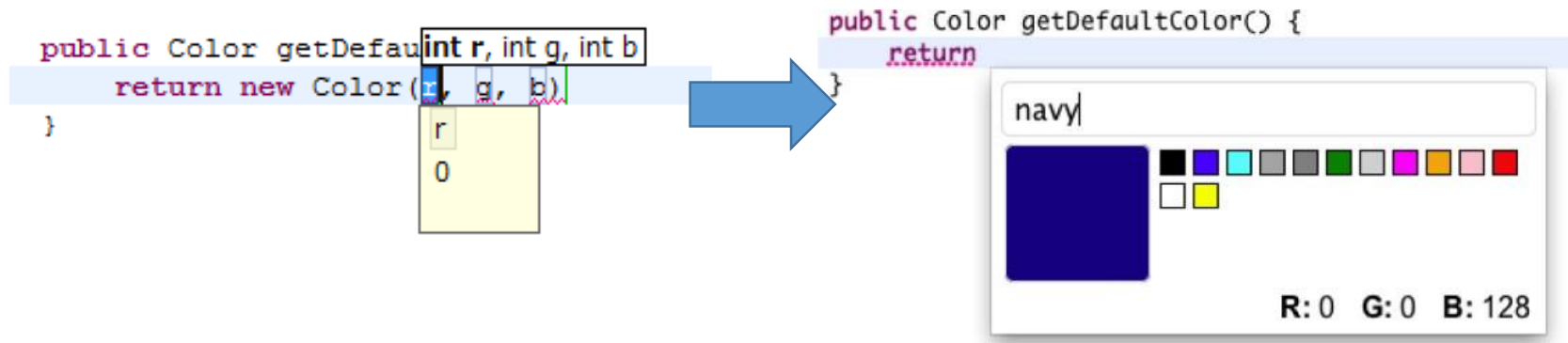
[Bragdon et al., CHI '10 & ICSE '10]



- **ファイル単位**のソースコード閲覧は文脈が飛んでつらい
- **関数の呼び出し階層**に沿って読み進められるようにした

# Active Code Completion

[Omar et al., ICSE '12]



- エディタのコード補完機能は常に文字ベースで、色など複雑なデータの指定に向いていなかった
- 補完すべきデータの種類(型)に応じて特別なインタフェースを表示して使いやすくした

# 視覚表現を活用した 統合開発環境

IDE with graphical representations

学部生講義「ユーザインタフェース」

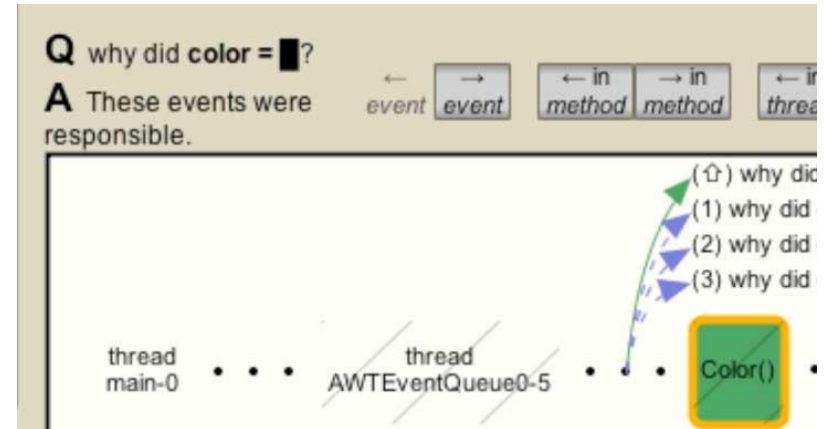
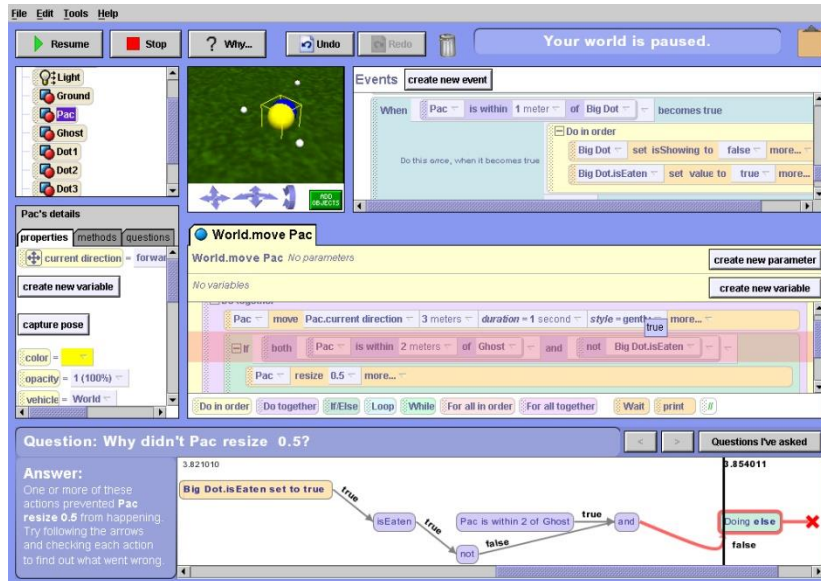
「Programming Environments (User Interface for Developers)」

# 視覚表現を活用した 統合開発環境

- **Whyline**: 画面表示の「なぜ」を解決
- **Barista**: エディタに視覚表現を活用
- **Sikuli**: 画像を貼りこめるエディタでGUI自動化
- **Picode**: 写真を貼りこめるエディタで姿勢情報処理
- **DejaVu**: 画像入力・処理・画面出力を簡単可視化

# Whyline

[Ko et al., CHI '04, ICSE '08 & CHI '09]



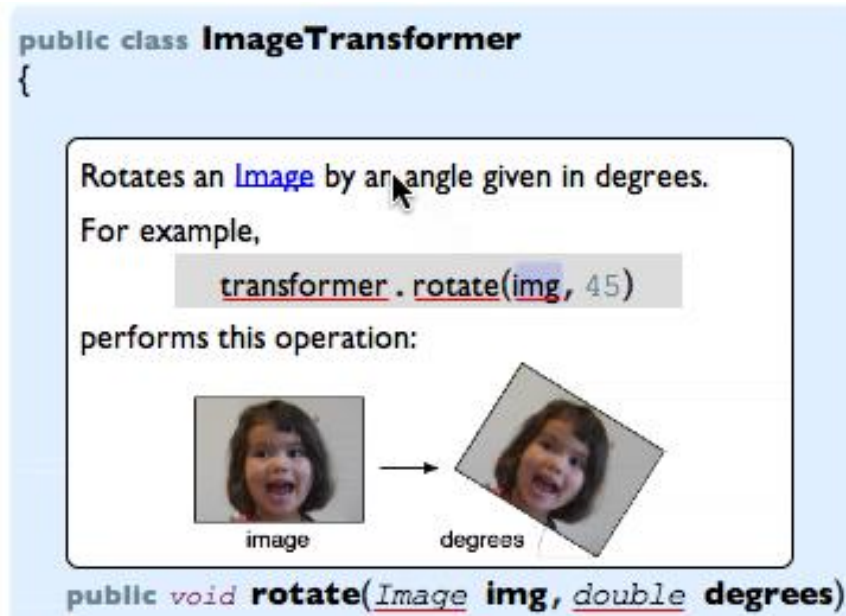
Whyline for Java

Whyline

- 画面出力が思い通りにならないことはよくある
- 「なぜ」現在の状態になっているのか、プログラムの実行過程を記録・分析して、原因の「行」を教えてくれる

# Barista


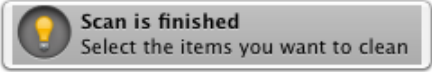

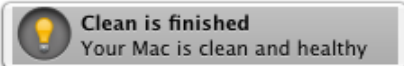
[Ko et al., CHI '04, ICSE '08 & CHI '09]



- 文字だけのソースコードは読みづらい
- 普通の文字ベースのエディタに見える構造化エディタを実装できるBaristaで視覚に訴えるソースコードエディタを実現

# Sikuli

[Yeh et al., UIST '09] [Chang et al., CHI '10]

```
switchApp("CleanMyMac.app")
click()
click()
while not find():
    sleep(5)
click()
while not find():
    sleep(5)
closeApp("CleanMyMac.app")
```

- 人対人なら「**これ**」と視覚的に指示できる内容がプログラムだと分かりにくい文字列になるのはおかしい
- **画面キャプチャ**を貼りこめる**ソースコードエディタ**と画像を引数に取れる**API**で、**GUI自動化スクリプト**が簡単に書ける




# Picode

[Kato et al., CHI '13]






## Kinectを使った人の姿勢比較

```
Pose pose = human.getPose();
```

```
if (pose.eq()) { /* do sth */ }
```

## LEGO Mindstormsの姿勢制御

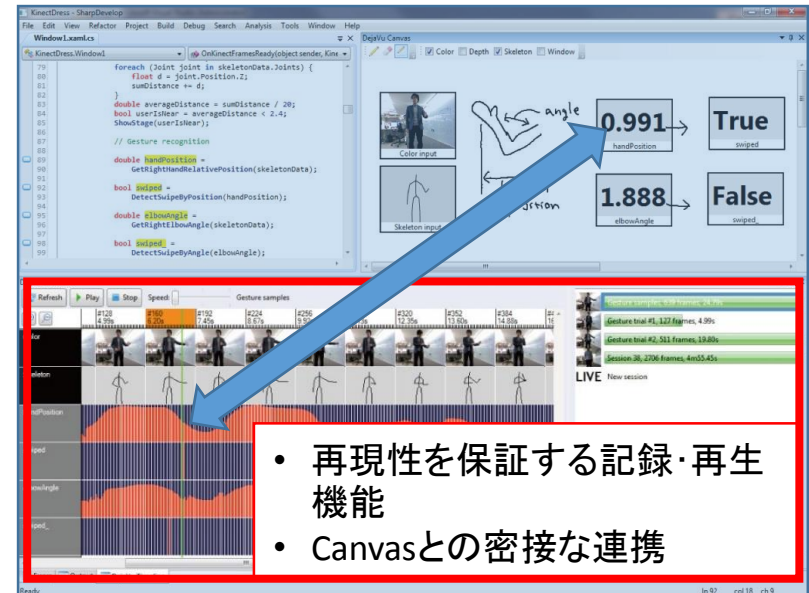
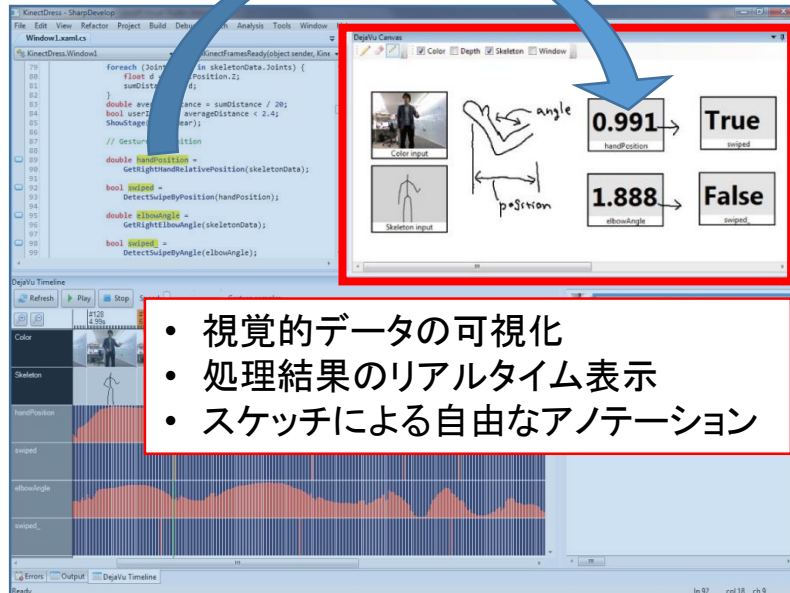
```
robot.setPose();  
Action a = robot.action();
```

```
a = a.pose() .wait(100) .pose();  
a.play();
```

- Sikuliの**実世界版** (姿勢データと写真を紐づける)
- **写真**を貼りこめる**ソースコードエディタ**と写真を引数に取れる**API**で、**姿勢情報処理プログラム**が簡単に書ける

# DejaVu

[Kato et al., UIST '12]



- **文字表現**しか扱えず、**ブレイクポイント**でしかメモリの中身を覗けないデバッガを改良
- **カメラ入力、変数、ウィンドウ出力**を記録・可視化・再生でき、プログラム起動中は表示が**リアルタイム**に更新される

# 特別な目的に最適化 された統合開発環境

IDE for specific purpose

学部生講義「ユーザインタフェース」

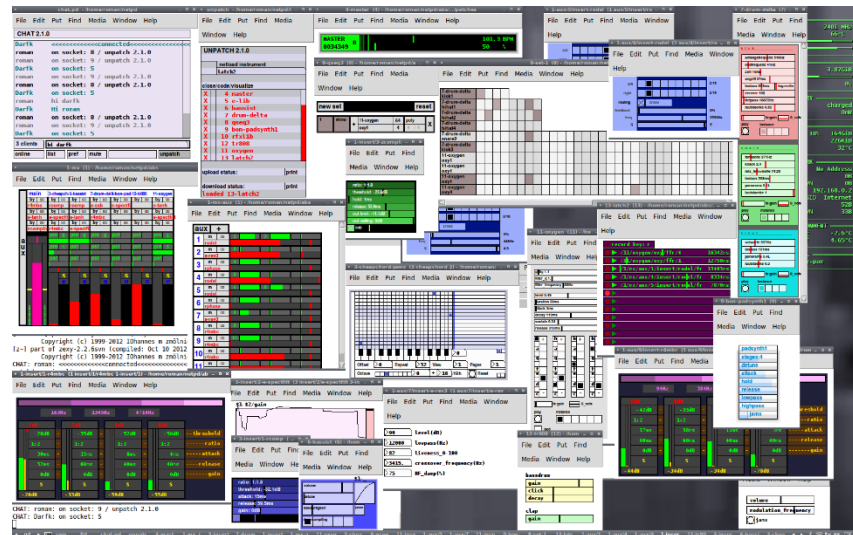
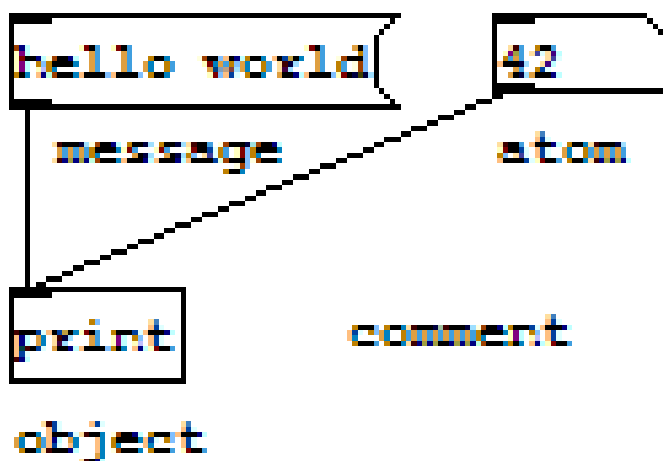
「Programming Environments (User Interface for Developers)」

# 特別な目的に最適化された 統合開発環境

- **PureData:** 音のLive Programming
- **Processing, Arduino:** メディアアート
- **d.tools:** Physical Computing
- **Gestalt:** 機械学習
- **Bret Victor's demo:** ゲームetc.のLive Programming

# PureData

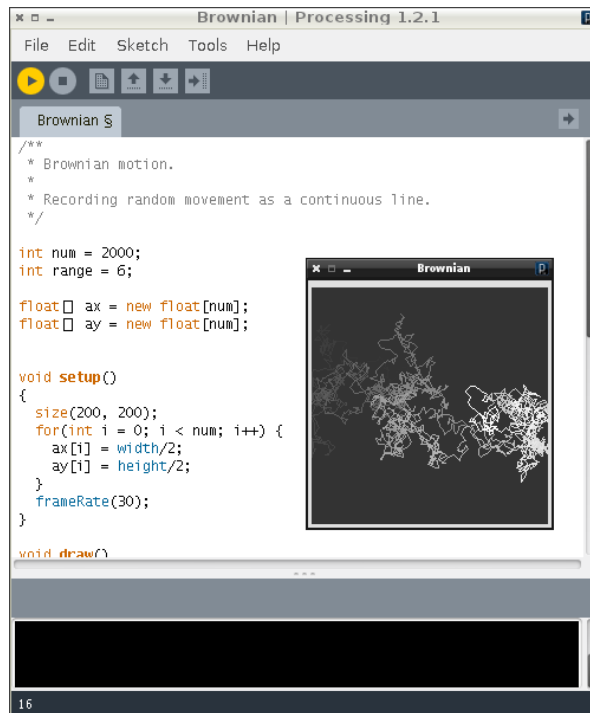
[Puckette et al., '96-]



- 音声信号処理のためのVisual Programming Language
- データ(message, atom)と関数(object, patch)を繋いでいくことで、音声・マルチメディア処理のプログラムが作れる
- 似たような商用アプリケーションのMax/MSPと同じ開発者

# Processing, Arduino

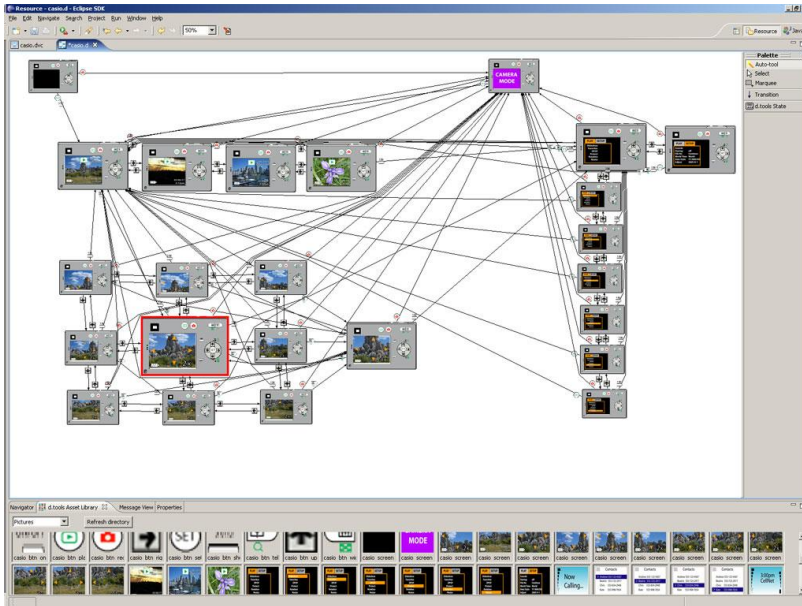
[Reas et al., '01-]



- それぞれメディアアート・マイコンプログラミング用の環境
- それぞれJava・Cベースの言語で簡単に開発できる

# d.tools

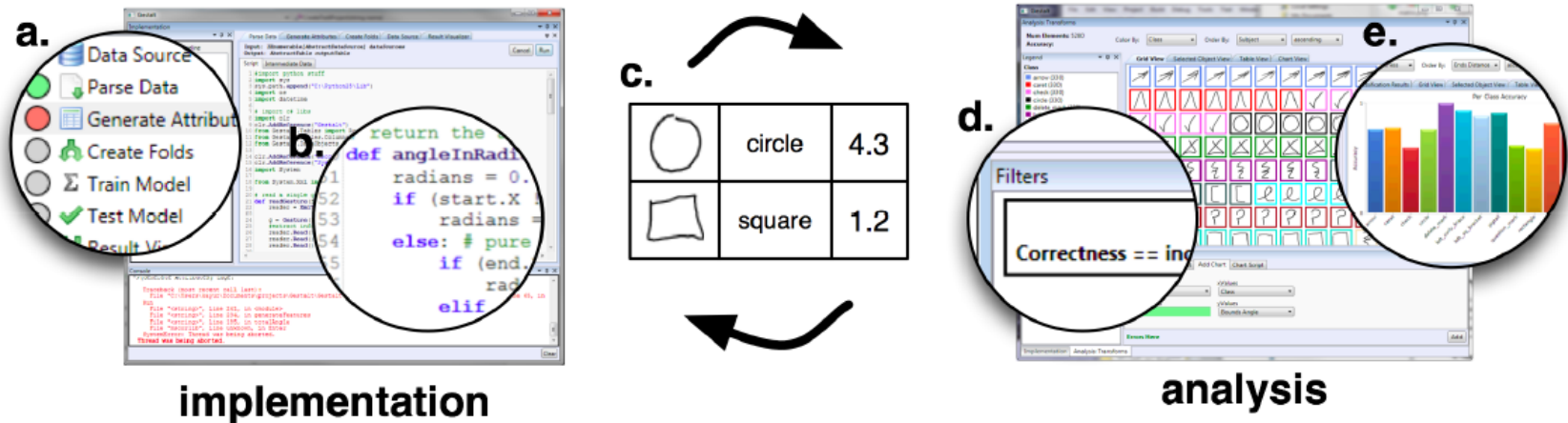
[Hartmann et al., UIST '06]



- Arduinoなどを使った**プロトタイピング**のための環境
- 状態遷移図をArduinoなどのバイナリにコンパイルできる
- 状態遷移と共に動画を記録する**ユーザテスト**用機能もある

# Gestalt

[Patel et al., UIST '10]



- 機械学習アルゴリズムの実装に最適化された環境
- サンプルの収集、閲覧、学習などが一手に行える



# 統合開発環境と インターネット

IDE and the Internet

学部生講義「ユーザインタフェース」

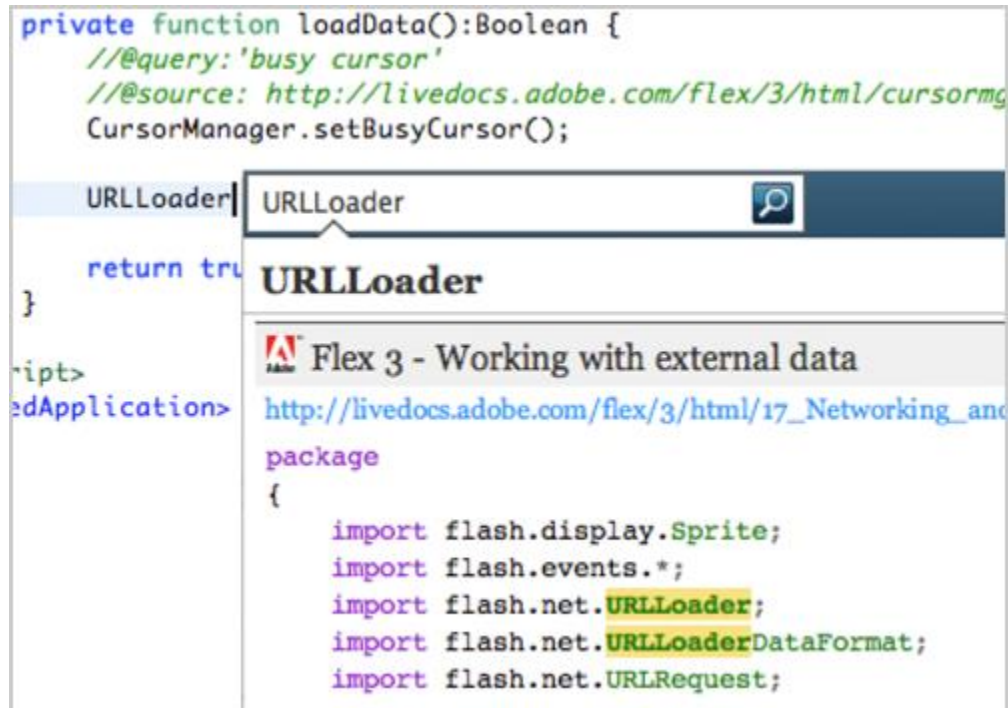
「Programming Environments (User Interface for Developers)」

# 統合開発環境とインターネット

- **Blueprint:** ネット上のサンプルコードを補完
- **HelpMeOut:** 典型的な例外の直し方を収集・提案
- **Collabode:** ネット介してペアプログラミング
- **TouchDevelop:** ブラウザ上の統合開発環境
- **MOOCs** (Massive open online course)

# Blueprint

[Brandt et al., CHI '10, CHI '12 & UIST '12]



- オンラインにたくさん**サンプルコード**があるのに使いづらい
- **コード補完**の一環でサンプルを検索して貼れるようにした

# HelpMeOut

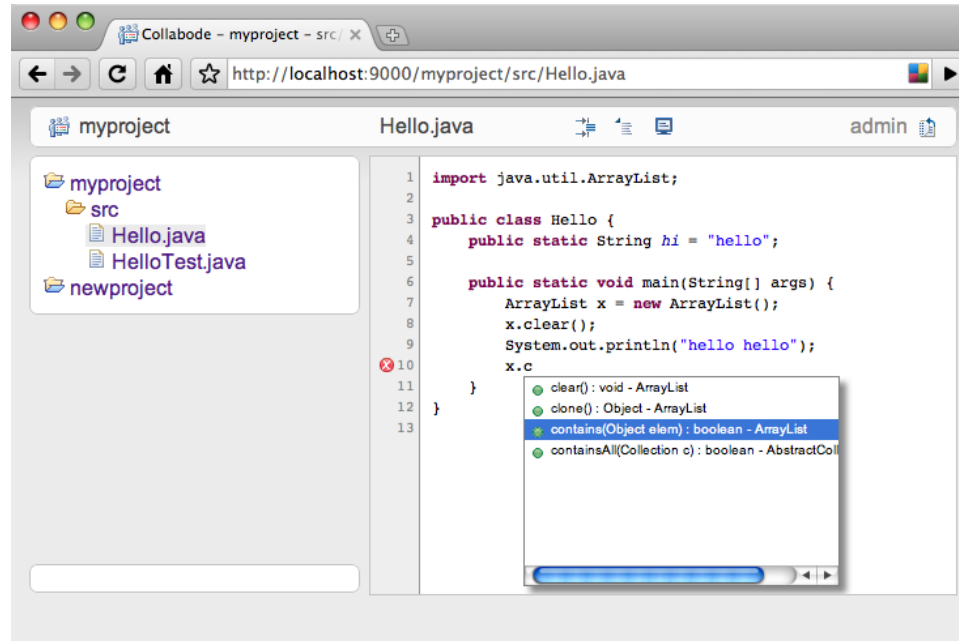
[Hartmann et al., CHI '10]



- **初心者**は同じようなバグでつまづきやすい
- **例外**が出たときの**典型的な解決策**を収集・推薦

# Collabode

[Goldman et al., UIST '11]



- **ペアプログラミング**に最適化された環境
- 一人がプログラムを書いてもう一人がテストを書く

# TouchDevelop

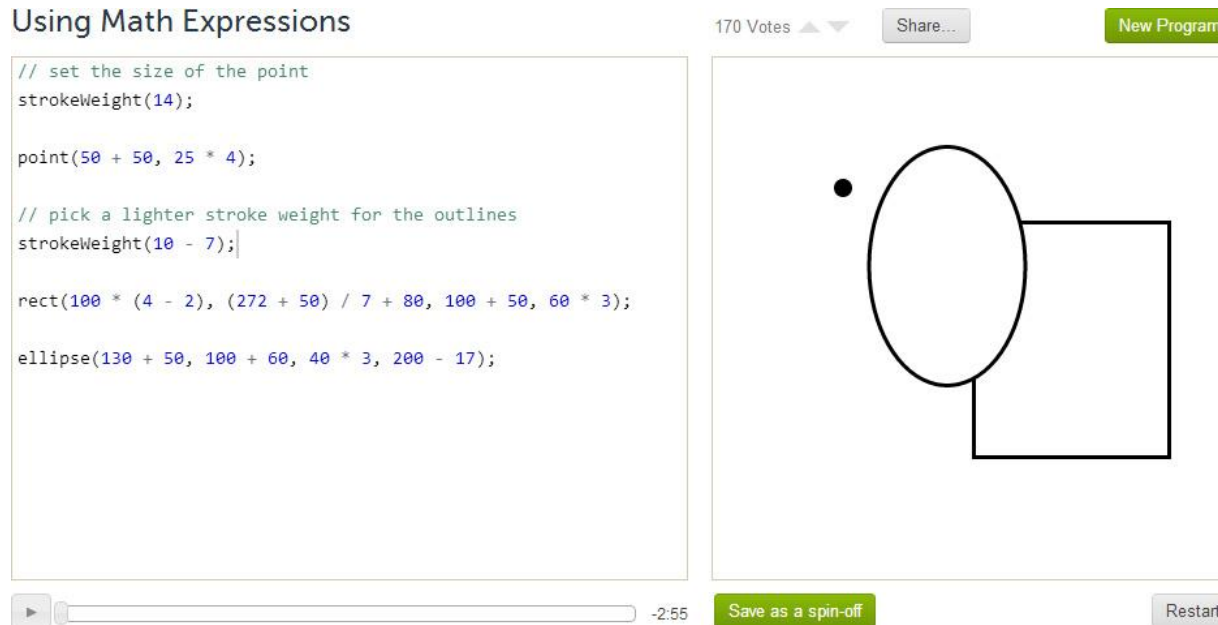
[Microsoft Research, '11-]



- **タッチデバイス**に最適化されたWebベースの環境
- 研究者がチームで開発しているのでいろいろな機能が詰め込んである
- 文字ベースだが構造化エディタのように選択肢を選ぶだけでプログラムが完成
- 全ユーザの利用頻度に応じた優先順位付きコード補完
- GUIをプログラム起動中に編集できるLive Programming

# MOOCs

## Massive Open Online Courses



- EdX MITx – 相互採点(Caesar)、自動採点システム
- Coursera – インタラクティブな講義資料兼用の環境

# 今回の内容

- プログラミングって何？

- 文字ベースの統合開発環境

Dartmouth BASIC, Visual Studio, Elipse, Code Bubbles, Active Code Completion

- 視覚表現を活用した統合開発環境

Whyline, Barista, Sikuli, Picode, DejaVu

- 特別な目的に最適化された統合開発環境

PureData, Processing, Arduino, d.tools, Gestalt

- 統合開発環境とインターネット

Blueprint, HelpMeOut, Collabode, TouchDevelop, MOOCs



# 付録

## Appendix

学部生講義「ユーザインタフェース」

「Programming Environments (User Interface for Developers)」

# その他の開発環境

- Bret Victor's demo (Inventing on Principles) [**Victor**, '12]
- HyperCard & Self [**Apple Computer**, '87-'04 & '95]

# Researchers

- Brad Myers, CMU HCII
- Rob Miller, MIT CSAIL
- James Landay, University of Washington
- Scott Klemmer, Stanford HCI
- Bjoern Hartmann, UC Berkley BiD
- Andrew Ko, University of Washington
- Joel Brandt, Adobe Research