

Direct Manipulation Programming



Sketch-n-Sketch



Ravi
Chugh



Nick
Collins



Brian
Hempel



Justin
Lubin



Mikaël
Mayer



THE UNIVERSITY OF
CHICAGO

ICFP Tutorial
09.27.2018

The screenshot shows the PyCharm IDE interface with the following details:

- Project Structure:** The project is named "djtp_first_steps" and contains files "tests.py", "models.py", and "admin.py".
- Code Editor:** The "tests.py" file is open, showing test cases for a "polls" application. One test case is currently being run, indicated by a red dot in the gutter.
- Search Bar:** A search bar at the top right is set to "Database" mode, showing results for "result".
- Database Inspector:** On the right, the "Database" tool window is open, connected to "Django default". It lists tables such as "auth_group", "auth_permission", "auth_user", etc., and provides a detailed view of the "django_admin_log" table.
- Debug Tools:** The bottom panel shows the "Debug" tools, including a "Debugger" tab with a stack trace, a "Console" tab, and a "Variables" tab where local variables like "startTime" and "maxDiff" are listed.
- Status Bar:** The bottom status bar shows "Tests Failed: 4 passed, 3 failed (4 minutes ago)" and other system information like "34:9 LF" and "Git: master".

The screenshot shows the PyCharm IDE interface with the following components:

- Project View:** Shows the project structure with files like `tests.py`, `models.py`, and `admin.py`.
- Code Editor:** Displays Python test code for a Django application. A search bar at the top right is set to "Search Everywhere" and contains the query "result". Below the search bar, a dropdown menu lists results from "Classes", "Files", "Symbols", and "Actions".
- Database:** Shows the "Django default" database schema with tables such as `auth_group`, `auth_permission`, `auth_user`, `auth_user_groups`, `auth_user_user_permissions`, and `django_admin_log`.
- Debug View:** The "Debug" tab is active, showing the "Debugger" tool window. It displays the current stack frame: `MainThread` (highlighted in yellow) and `test_index_view_with_a_future_question` (highlighted in blue). The "Variables" pane shows local variables: `longMessage` (bool, False), `maxDiff` (int, 640), `reset_sequences` (bool, False), `serialized_rollback` (bool, False), and `startTime` (datetime, 2015-10-09 11:38:35.521452). The "Watches" pane shows a watch for `self.maxDiff` (int, 640).
- Test Results:** The bottom status bar indicates "Tests Failed: 4 passed, 3 failed (4 minutes ago)".

t

+





Sketch-n-Sketch

Sketch-n-Sketch file:///Users/ravi/git-clones/github-ravichugh/sketch-n-sketch/build/out/index.html Sketch-n-Sketch File Code Tools View Options ▲ Previous Example Next Example

Current file: Untitled (Get Started) *

Undo Redo Run ▶

```
1 type LogoParams
2   = TopLeft {x:Num, y:Num, size:Num}
3   | Center {cx:Num, cy:Num, rad:Num}
4
5 logo : String -> LogoParams -> Svg
6 logo fill logoParams =
7   let
8     {x, y, size} =
9       case logoParams of
10         TopLeft data ->
11           data
12         Center {cx, cy, rad} ->
13           {x=cx-rad, y=cy-rad, size=2*rad}
14
15   (cx, cy) =
16     (x + 0.5*size, y + 0.5*size)
17   in
18     [ rect fill x y size size
19     , line "white" 10 x y (x + size) (y + size)
20     , line "white" 10 x (y + size) cx cy
21     ]
22
23 main =
24   svg <| logo "gray" <| TopLeft {x=100, y=130, size=200}
```

Raw Stretchy Sticky

5

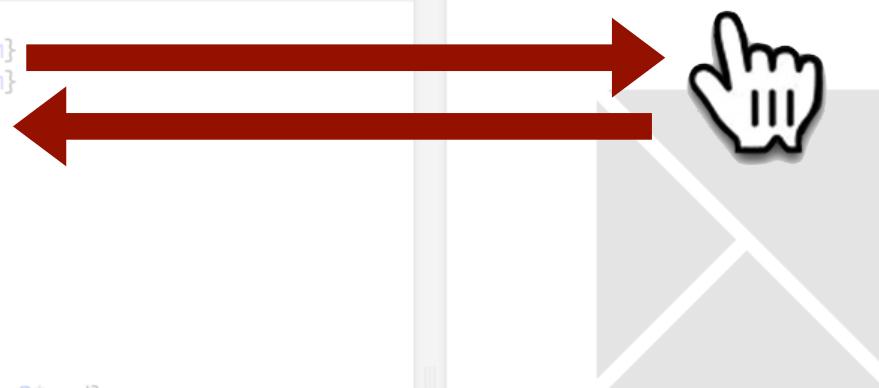


Sketch-n-Sketch

[PLDI 2016] [OOPSLA 2018]

Bidirectional Programming

```
1 type LogoParams
2   = TopLeft {x:Num, y:Num, size:Num}
3   | Center {cx:Num, cy:Num, rad:Num}
4
5 logo : String -> LogoParams -> Svg
6 logo fill logoParams =
7 let
8   {x, y, size} =
9     case logoParams of
10       TopLeft data ->
11         data
12       Center {cx, cy, rad} ->
13         {x=cx-rad, y=cy-rad, size=2*rad}
14
15   (cx, cy) =
16     (x + 0.5*size, y + 0.5*size)
17 in
18   [ rect fill x y size size
19   , line "white" 10 x y (x + size) (y + size)
20   , line "white" 10 x (y + size) cx cy
21   ]
22
23 main =
24   svg <| logo "gray" <| TopLeft {x=100, y=130, size=200}
```



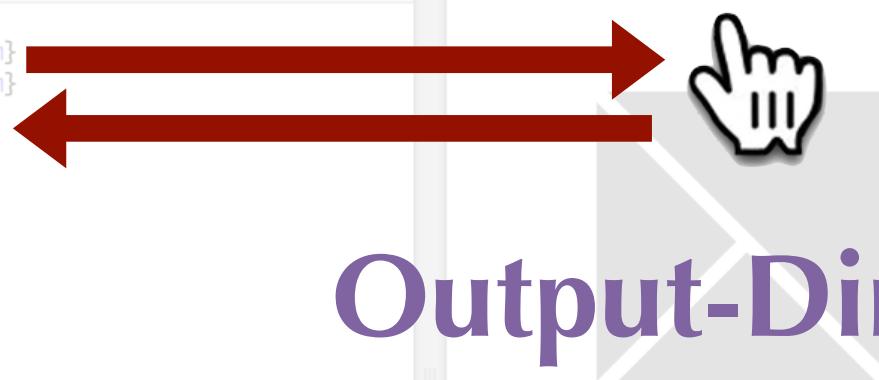


Sketch-n-Sketch

[PLDI 2016] [OOPSLA 2018]

Bidirectional Programming

```
1 type LogoParams
2   = TopLeft {x:Num, y:Num, size:Num}
3   | Center {cx:Num, cy:Num, rad:Num}
4
5 logo : String -> LogoParams -> Svg
6 logo fill logoParams =
7 let
8   {x, y, size} =
9     case logoParams of
10       TopLeft data ->
11         data
12       Center {cx, cy, rad} ->
13         {x=cx-rad, y=cy-rad, size=2*rad}
14
15   (cx, cy) =
16     (x + 0.5*size, y + 0.5*size)
17 in
18   [ rect fill x y size size
19   , line "white" 10 x y (x + size) (y + size)
20   , line "white" 10 x (y + size) cx cy
21   ]
22
23 main =
24   svg <| logo "gray" <| TopLeft {x=100, y=130, size=200}
```



Output-Directed Programming

[UIST 2016] [wip]



Sketch-n-Sketch

[PLDI 2016] [OOPSLA 2018]

Bidirectional Programming

```
1 type LogoParams
2   = TopLeft {x:Num, y:Num, size:Num}
3   | Center {cx:Num, cy:Num, rad:Num}
4
5 logo : String -> LogoParams -> Svg
6 logo fill logoParams =
7 let
8   {x, y, size} =
9   case logoParams of
10     TopLeft data
11       data
12     Center {cx, cy, rad} -
13       {x=cx-rad, y=cy, rad=size+rad}
14
15   (cx, cy) -
16   (x + 0.5*size, y + 0.5*size)
17 in
18   [ rect fill "white" x y size size
19     , line "black" "10px" x y (x + size) (y + size)
20     , line "black" "10px" y (x + size) cx cy
21   ]
22
23 main =
24   svg <! logo "gray" > topLeft {x=100, y=130, size=200}
```

Structured
Text Editing

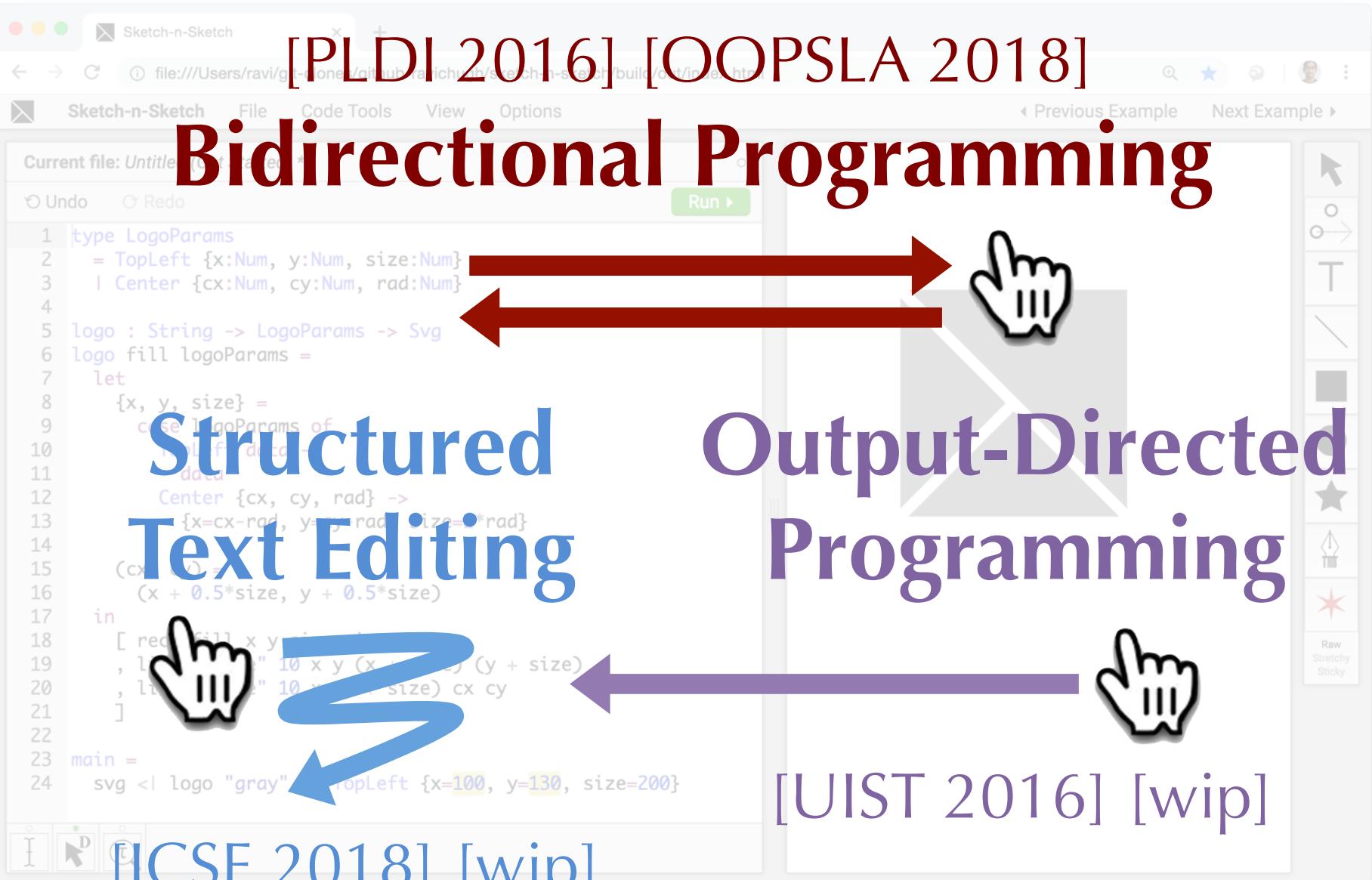


Output-Directed
Programming



[UIST 2016] [wip]

[ICSE 2018] [wip]





Sketch-n-Sketch

Sketch-n-Sketch file:///Users/ravi/git-clones/github-ravichugh/sketch-n-sketch/build/out/index.html Sketch-n-Sketch File Code Tools View < Previous Example Next Example Current file: Untitled (Get Started) *

Undo Redo

```
1 type LogoParams
2   = TopLeft {x:Num, y:Num, size:Num}
3   | Center {cx:Num, cy:Num, rad:Num}
4
5 logo : String -> LogoParams -> Svg
6 logo fill logoParams =
7 let
8   {x, y, size} =
9     case logoParams of
10       TopLeft data ->
11         data
12       Center {cx, cy, rad} ->
13         {x=cx-rad, y=cy-rad, size=
14           (cx, cy) =
15             (x + 0.5*size, y + 0.5*size)
16         in
17           [ rect fill x y size size
18             , line "white" 10 x y (x + size)
19             , line "white" 10 x (y + size) cx
20           ]
21         ]
22
23 main =
24   svg <! logo "gray" <! TopLeft {x=100, y=130, size=200}
```

A portrait photograph of a young man with curly brown hair, smiling. He is wearing a red cable-knit sweater. To his right is a gray rectangular area containing a white diagonal 'X' shape.Icons for selection, move, delete, and zoom.

Justin Lubin
~now–1:15pm



Sketch-n-Sketch

[PLDI 2016] [OOPSLA 2018]

Bidirectional Programming



Mikaël Mayer
~1:15–3:10pm



2:10pm
3:10pm





A screenshot of the Sketch-n-Sketch application interface. On the left, there is a large preview window showing a portrait of a man with short brown hair, identified as Brian Hempel. To the right of the preview is a code editor window displaying the following pseudocode:

```
1  ty
2
3
4
5  lo
6  lo
7
8
9
10
11
12
13
14
15
16
17
18
19
20 , line "white" 10 x (y + size) cx cy
21
22 main =
23   svg & logo "gray" & TopLeft {x=100, y=130, size=200}
```

Below the code editor, the text "Brian Hempel" and "3:30-3:45pm" is displayed. The right side of the slide features a large purple arrow pointing from the text area towards the code editor. To the right of the arrow is a large purple hand icon pointing upwards. The text "Output-Directed Programming" is written in large purple letters across the slide. At the bottom right, the text "[UIST 2016] [wip]" is shown in purple.



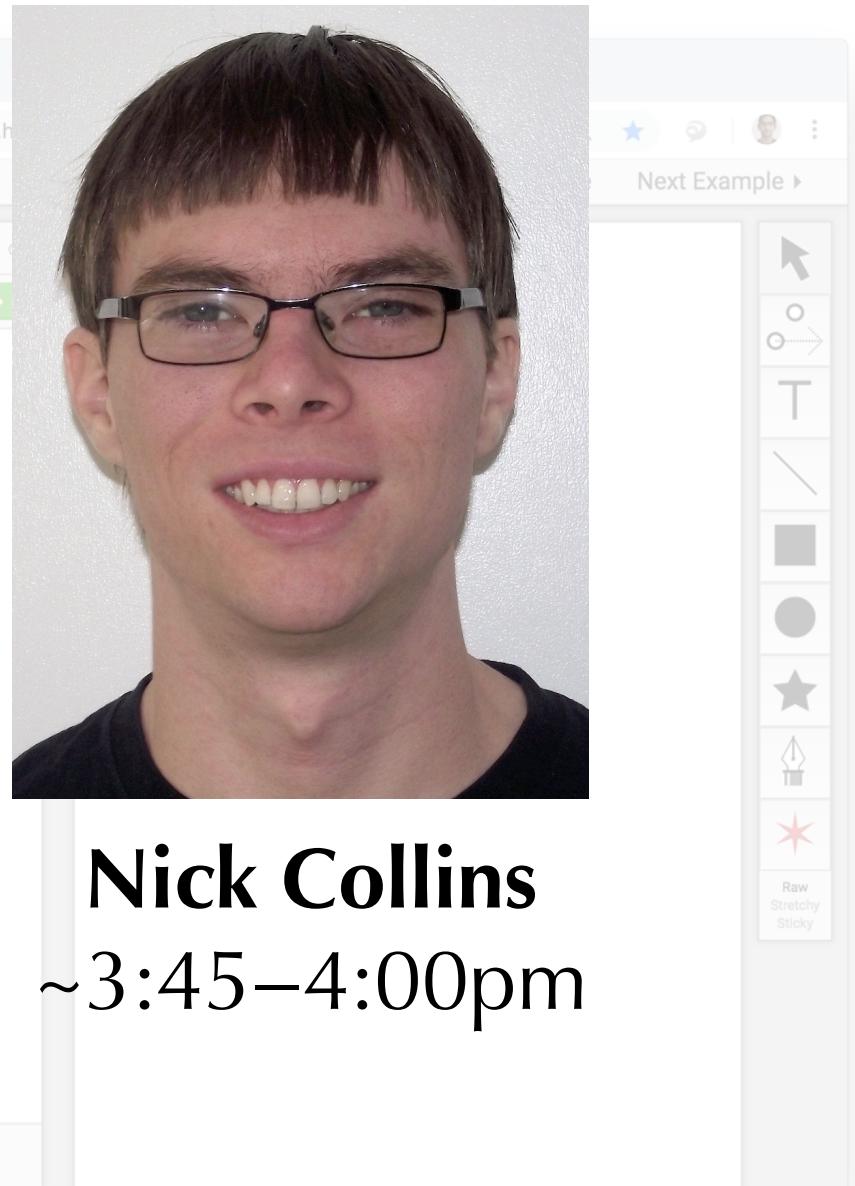
Sketch-n-Sketch

The screenshot shows the Sketch-n-Sketch application window. On the left is a code editor with the following TypeScript code:

```
1 type LogoParams
2   = TopLeft {x:Num, y:Num, size:Num}
3   | Center {cx:Num, cy:Num, rad:Num}
4
5 logo : String -> LogoParams -> Svg
6 logo fill logoParams =
7 let
8   {x, y, size} =
9   case logoParams of
10     TopLeft data
11       data
12         Center {cx, cy, rad} ->
13           {x=cx-rad, y=cy, rad=rad-size}
14           (cx, cy)
15           (x + 0.5*size, y + 0.5*size)
16     in
17       [ rectFill x y size
18         , line "10px y (x + rad) (y + size)
19         , line "10px y size) cx cy
20       ]
21     ]
22
23 main =
24   svg <! logo "gray" > topLeft {x=100, y=130, size=200}
```

A large blue hand icon with a pointing finger is overlaid on the code editor, pointing to the line `size=200`. A blue wavy arrow originates from this line and points towards the preview area. In the bottom right corner of the code editor, the text "[ICSE 2018] [wip]" is displayed in blue.

The interface includes a toolbar at the top with icons for file operations, a menu bar with "Sketch-n-Sketch", "File", "Code Tools", "View", and "Options", and a "Run" button. The preview area on the right shows a portrait of a young man with glasses and a smile. A vertical toolbar on the right contains icons for various drawing tools like selection, move, rotate, text, line, rectangle, circle, star, and brush, along with "Raw", "Stretchy", and "Sticky" options.



The diagram illustrates the relationship between two paradigms: Structured Text Editing and Output-Directed Programming.

Structured Text Editing (blue text) is represented by a large blue arrow pointing from the left towards the central code editor. It is associated with the **[ICSE 2018] [wip]** label at the bottom left.

Output-Directed Programming (purple text) is represented by a large purple arrow pointing from the right towards the central code editor. It is associated with the **[UIST 2016] [wip]** label at the bottom right.

Bidirectional Programming (red text) is represented by a large red arrow pointing from the top towards the central code editor. It is associated with the **[PLDI 2016] [OOPSLA 2018]** label at the top center.

The central code editor displays a snippet of Haskell-like code for generating a logo:

```
1 type LogoParams
2   = TopLeft {x:Num, y:Num, size:Num}
3   | Center {cx:Num, cy:Num, rad:Num}
4
5 logo : String -> LogoParams -> Svg
6 logo fill logoParams =
7 let
8   {x, y, size} =
9   case logoParams of
10     TopLeft data
11       data
12     Center {cx, cy, rad} ->
13       {x=cx-rad, y=cy, rad=size-rad}
14
15 (cx, cy)
16   (x + 0.5*size, y + 0.5*size)
17 in
18   [ rect fill "white" x y size size
19     , line "black" "10px" (x + size) (y + size)
20     , line "black" "10px" size cx cy
21   ]
22
23 main =
24   svg <! logo "gray" > topLeft {x=100, y=130, size=200}
```

The code editor interface includes a toolbar with various icons (e.g., Undo, Redo, Run button), a status bar showing the current file, and a sidebar with tool icons.