Fixel

Kavita Jain-Cocks, Amrita Mazumdar, Darien Nurse, Nilkanth Patel, Matthew Patey

Team Roles

- Language & Tools Guru: Amrita Mazumdar
- System Architect: Darien Nurse
- Project Manager: Kavita Jain-Cocks
- Tester and Validator: Matthew Patey
- System Integrator: Nilkanth Patel

Overview

twitter

Pinterest

Why Fixel?

Customizable

Portable

User Friendly

Robust

Photo Sharing

Organizational & Development Tools

PLY

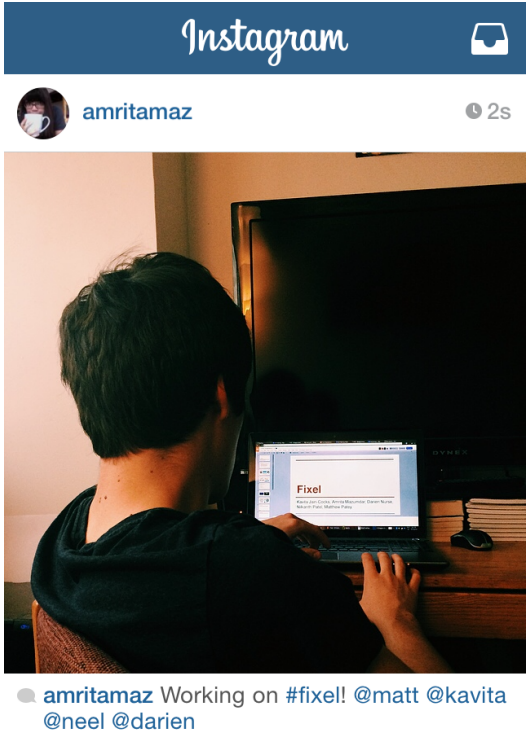
Jenkins

A Vintage Example

Before

After

Social-media inspired syntax




```
#myfilter @image0, 30

myfilter @imageName, @intensity:
    #grayscale @imageName
    #contrast @imageName, @intensity
```

- “hashtag” syntax for functions
- use ‘@’ to address variables

Image Processing Functionality

- Access images using @image0
- Lists of images @images[2]
- Pixel and color types
- Access color at specific image pixel using bracket accessors
- @colormy = @image0[30,20]

Forp - fixel's for loop for pixels

```
forp @pixel in @image0:
    @pixel = 255 - @pixel.color
------------------------------
-
invert.fxl
```

Compiling and Running Programs

```
> fixel program_name.fxl [image names]
```

- Invoking “fixel” translates and runs the Fixel program
- Translated program invokes built-in functions

Fixel is Easy to Write

```
#vintage @image0, 2

vintage @imageName, @size:
	#scale @imageName, @size
	#cropit @imageName, 30, 30,
(@imageName.width - 30), (@imageName.
height - 30)
	#grayscale @imageName
	#contrast @imageName, 80
	#overlay @imageName, (#color
"burlywood"), 30
	#border @imageName, 50, (#color
"burlywood")
```

Vintage Filter in Fixel

```
import os
import sys

# add fixel top to path so fixel functions can be imported
sys.path.append(os.path.abspath('/Users/amritamaz1/fixel/src/translator/..'))

from runtime import fixelFunctions
from runtime import runtime_classes

def vintage(imageName, size):
		fixelFunctions.scale(imageName, size)
		fixelFunctions.cropit(imageName, 30, 30, (imageName.width - 30), (imageName.height - 30))
		fixelFunctions.grayscale(imageName)
		fixelFunctions.contrast(imageName, 80)
		fixelFunctions.overlay(imageName, (fixelFunctions.color("burlywood")), 30)
		fixelFunctions.border(imageName, 50, (fixelFunctions.color("burlywood")))

inputImages = sys.argv[1:]
if len(inputImages) < 1:
		print "\nNo images were used as arguments. Please append the paths to the images you'd like to use as arguments
and run this Fixel program again.\n"
		sys.exit(0)
inputImageCount = 0
Namespace = type('Namespace', (object,), {'images': []})  # cleaner than having to declare a class
ns = Namespace()

# create variables for each image
for currentImage in inputImages:
		image = runtime_classes.Image(currentImage)
		setattr(ns, "image"+str(inputImageCount), image)
		ns.images.append(image)
		inputImageCount += 1

vintage(ns.image0, 2)

for image in ns.images:
		fixelFunctions.saveImage(image, "JPEG")
```

Vintage Filter in Python

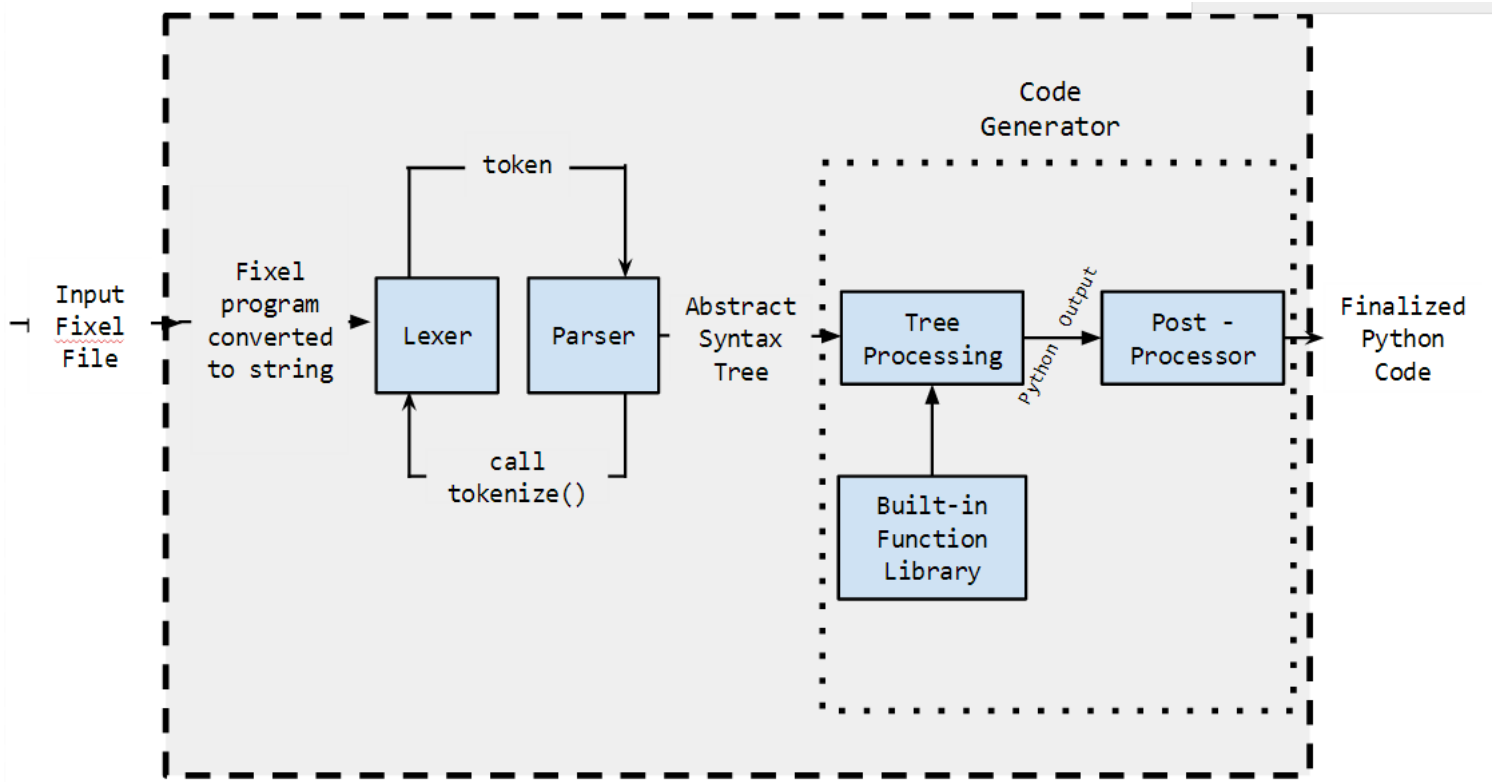
Built-in Functions

- Designed to give users something to start with.
- Building blocks to generate custom functions that can do a bunch of transformations.
- Written in Python, using PIL.
- Implementation is very simple, using the hashtag scheme to call them on images.

Example: Collage

```
#collage @image3, @images, 1600, 1200
```

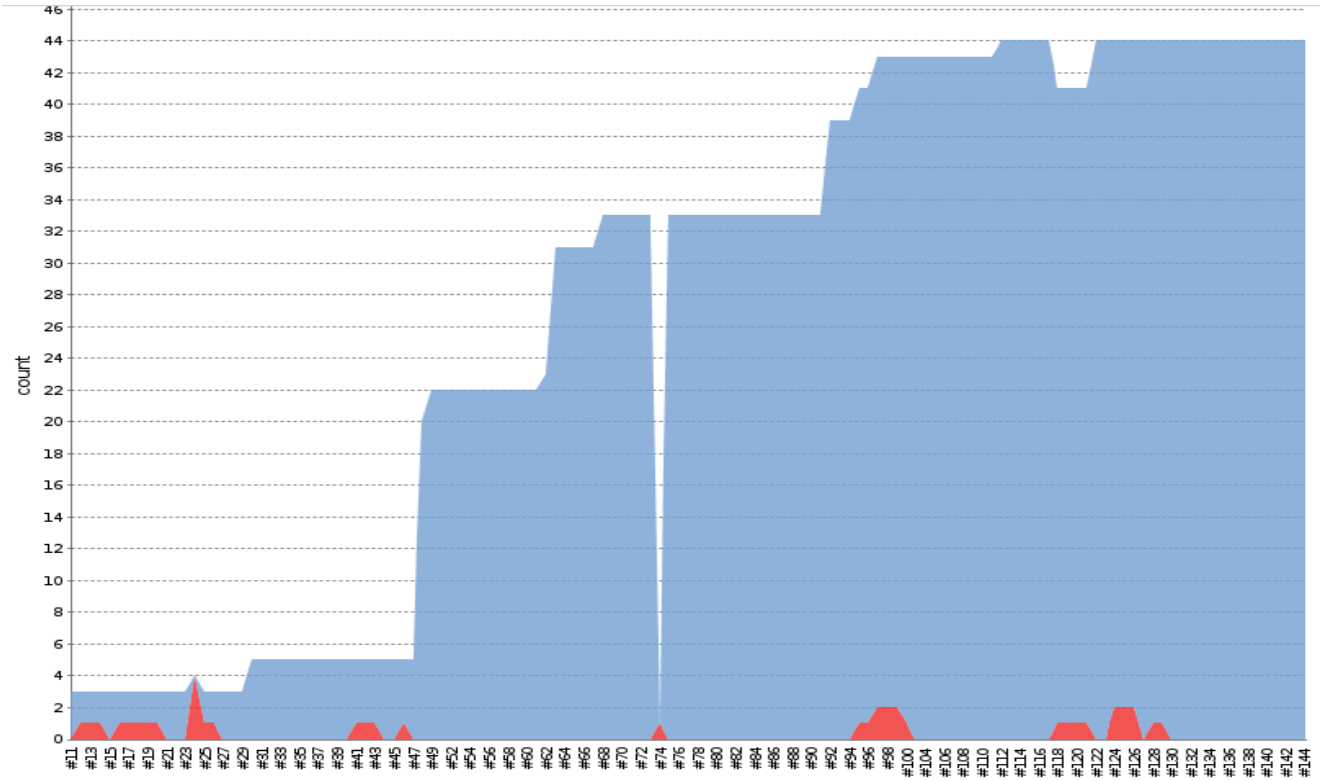
System Architecture



Testing

- Unit tests for lexer, parser, generator
  - isolate each unit, provide input, check output
  - fixel, tokens, trees, python
- Runtime tests for built in functions and data types
- Coverage used to ensure all important features covered
  - e.g. all possible tokens, grammar productions

Continuous Integration - Jenkins



Challenges & Obstacles

- Tokenizing Indentation
  - Sending multiple tokens on one regex match
  - Took Advantage of PLY Lexing states
- Declaring variables for the user
  - All local variables for main are attributes of an object
  - Dynamically set image variable attributes
- Forp
  - How to get variable assignment to update field of another object

The Vintage Example Revisited

Before

After

The Vintage Example Revisited

```
#vintage @image0, 2

vintage @imageName, @size:
    #scale @imageName, @size
    @picwidth = @imageName.width
    @picheight = @imageName.height
    #cropit @imageName, 30, 30, (@picwidth - 30), (@picheight - 30)
    #grayscale @imageName
    #contrast @imageName, 80
    #overlay @imageName, (#color "burlywood"), 30
    #border @imageName, 50, (#color "burlywood")
```

The Vintage Example Revsited

Before
Dimensions: 1725x1024

After
Dimensions: 3490x2088

Next Steps

Additional Build-in Function and Data Types

- Floating-point types and arithmetic
- Multi-line comments
- Conditional pixel for loops

Integration with Social Media

- Facebook, Instagram, Twitter, etc…

Interactive GUI

- View pictures as you code
- Preview images before saving

Robust Error Handling

What We Learned

- Decide on the small things early (when to typecheck, variable declarations, syntax for calling functions).
- Get a hello world function running first before trying to have the entire grammar implemented.
- Have group members always working on the same sort of assignments so that they become “experts” on those aspects of the language.

Thank you!

Any questions?