# Web Scraping with Nokogiri/Kimono

Robert Krabek
CS50 Fall 2015

## Workspace setup and installation

Creating a new workspace

- CS50 IDE > Go To Your Dashboard
- Create a new workspace > Custom Template > Create Workspace
- `$ gem install nokogiri`

On a current or CS50 template workspace

- `$ CFLAGS= gem install nokogiri`

www.nokogiri.org

# Ruby Syntax

Declare variables with no type, no semicolons (yay!)

No parentheses, put ‘end’ after a block is finished

Only += and -= no ++ or --

require ‘some_library’ instead of #include

Variable sized arrays, you can append with <<

No chars, just 1 letter strings

## Basic Nokogiri Scrape Setup

Include required libraries

Open the url and store into a variable

Search variable for unique html tag with .css

Output content

## File I/O in Ruby

Similar to C

```
fname = "description.txt"

file = File.open(fname, "w")

file.puts description

file.close
```

Can also use csv and json libraries to output csv and json files

## Using Kimono

Navigate to page of interest

Click on field of interest

Calibrate field detection

Create API

## Pros and Cons

Nokogiri

- Fast, easy to test, easy to configure, no page limits
- Can only scrape HTML

Kimono

- Easy to use, can scrape javascript,
- Max 25 pages at a time, hard to configure/set up nested scrapes

## Alternatives

Python

- beautiful soup (similar to nokogiri), PyQT (dynamic)

Ruby

- Capybara/Poltergeist (dynamic)

Javascript

- PhantomJS(dynamic)

Selenium browser automation