This class works differently from (probably) any class you've ever taken before.

- Class time is for learning the big, important ideas. You'll be reading a lot of code.
- Homework time is for learning the details. You'll be writing a lot of code.

- Notes like this will be provided for every class. So don't worry about writing everything down.
- When I'm demonstrating something, don't take more notes. You'll miss something. **Your job is to think of questions.**
- You are free to jot down extra notes during the labs, during break times, and after class.
- You *will* need to look up more details online. I provide helpful links on the course website. Oh and there's a thing now called Google.

- The code that I write in class is always available for review.
- **99% of what you'll need to solve the homework problems will be in my code.**

Consider The Following…

Professional developers make their money by knowing how to invent their own solutions through research, not by memorizing specific solutions.

They have learned how to think. Not how to copy.

- When you Google: try to use specific, uncommon words; or, paste an entire error message.
- Filter results to only the past year or shorter (click Search Tools on the right side).
- You can trust answers from rubyonrails.org, ruby-lang.org, and stackoverflow.com.
- Read error messages.
- Watch out for *singular vs. plural* words: movie vs. movies, user vs. users, etc.
- Make sure that blog or tutorial you're reading is for Rails 4+ and Ruby 2+
- **Don't believe online tutorials** (using code generators, etc.) until after this course.
- Just Try It™. You will learn more by experimenting than by thinking or searching or reading.
- Write as much code as you can. Make up tiny problems for yourself and try to solve them.
- Watch your Rails server log.
- Read error messages.
- Always look for cause and effect. There is a reason for everything.
- Got code that works but not sure why? Intentionally break the code, and see what changes.
- Be curious about everything.
- Stuck for more than five minutes? Ask the ducky™.
- What's fuzzy? What's crystal-clear?
- Read error messages.

Consider The Following…

If you want to work in companies that create digital products, this course is a great start, but you'll have to keep learning more when this course is over.

What should you learn now so that you can keep learning when it's over?

- Real-world things represented as virtual things in your application are called *models*.
- Collectively, all of the models in our application, as well as the relationships between them, are known as your application's *domain model*.
- Models have *data* associated with them. Our job is to figure out what data we need to store. These data points are called *attributes*. In the following, our model is called Movie, and our attributes are id, title, year, and actors.
- **Models always have a numeric "ID" attribute.**

Movie

id	title	year	actors	studio
1	Apollo 13	1995	Tom Hanks, Bill Paxton	Paramount
2	Guardians of the Galaxy	2014	Chris Pratt, Zoe Saldana	Marvel
3	Backdraft	1991	Kurt Russell, William Baldwin	Fox
4	Star Wars	1977	Mark Hamill, Harrison Ford	Lucasfilm
5	Toy Story	1995	Tom Hanks, Tim Allen	Pixar

Consider The Following…

What if we want to store more data about actors? What if an studio ever changed their name? Would we run out of room if we tried to store all the actors in the movie?

- Your application can have as many models as you want.
- Models can have relationships between them.
- There are two techniques for relating models.
  - One-to-many (One studio has many Movies, a Movie belongs to only one Studio)
  - Many-to-many (Actors have Roles in many Movies, a Movie has many Actors)
- **We implement relationships in data by using the numeric ID field.**

Movie			
id	**title**	**year**	**studio id**
1	**Apollo 13**	**1995**	**1**
2	**Star Wars**	**1977**	**2**
3	**Toy Story**	**1995**	**3**
4	**Raiders of the Lost Ark**	**1981**	**2**

Studio	
id	**name**
1	**Paramount**
2	**Lucasfilm**
3	**Pixar**

Actor	
id	**name**
1	**Tom Hanks**
2	**Bill Paxton**
3	**Harrison Ford**

Role		
id	**movie_id**	**actor_id**
1	**1**	**1**
2	**1**	**2**
3	**2**	**3**
4	**3**	**1**
5	**4**	**3**

- Use **rails new** to generate a new Rails application
- **A Rails application is just a folder with a bunch of files in it.**
- If you clone an existing app from GitHub, run **bundle install**
- Use **rails server** to start a local HTTP server on http://localhost:3000
- Press Ctrl-C to stop the server
- Learn the ***RCAV™*** recipe for creating web pages in Rails: **R**oute-**C**ontroller-**A**ction-**V**iew
- **Routes** are defined in *config/routes.rb*
- **Controllers** are files in *app/controllers*
- **Actions** are methods (stuff between *def…end*) defined in the controller
- **Views** are files in the folder named after the controller, e.g. *app/views/movies/* and can be plain HTML or HTML with embedded Ruby (.erb)

config/routes.rb

```
get '/flicks', to: 'movies#index'
```

app/controllers/movies_controller.rb

```
class MoviesController < ApplicationController

  def index
  end

end
```

app/views/movies/index.html

```
<html>
<body>
…
</body>
</html>
```