Ruby on Rails

CSE 190M, Spring 2009

Week 8
Models

• Up to this point, all of our work has been in our Controllers and View

• If you have inspected the Models created by the Scaffolding, you will notice that they are empty
  – But they inherit from ActiveRecord::Base
  – This is what gives us access to the fields (methods) of the objects as they are in the database without defining the methods ourselves

• So... why do we even have models?
Models

• We use Models define methods that provide information about individual objects
• These methods usually do calculations on other properties of the object that you will not want to write over and over again
• Example:
  
  user.admin?  # checks if a user is a admin user
  user.authenticate  # authenticates a user
  gallery.empty?  # checks if a gallery is empty
  gallery.clear  # removes all the gallery images
• Often, there are inherit relationships between the different object we are creating
  – In a blog
    • Users have many Entries; an Entry belongs to only one User
    • Entries have many Comments, and a Comment belongs to only one Entry
  – In a login system
    • Users have many Roles; Roles belong to many Users
  – In a course registration system
    • A Student has many courses; a course has many students
Types of Relationships

• One-to-One
  – A U.S. citizen has only one S.S.N; Each S.S.N. belongs to only one U.S. citizen

• One-to-Many
  – A person owns many cars; A car belongs to only one owner
  – A company has many employees; An employee is employed by only one company

• Many-to-Many
  – A student has many courses; A course has many students
  – A programmer has many projects; A project has many programmers
  – A blog post has many posters; A poster has many posts
Relationships in Models

• One-to-One
  – has_one/belongs_to

• One-to-Many (most common)
  – has_many/belongs_to

• Many-to-Many
  – has_many/has_many
  – has_and_belongs_to_many
  – These are tricky… So we will not go into detail with these

• An object that belongs_to another should reference that object by id in the database
  employee.company_id
In Our Gallery Class

```ruby
class Gallery < ActiveRecord::Base
  has_many :images # plural: images
end
```

In Our Image Class

```ruby
class Image < ActiveRecord::Base
  belongs_to :gallery # singular: gallery
end
```