ACTIVE RECORD INHERITANCE (ARI)

Oop Basics

1. **What is Active Record ?**
2. **What is Inheritance ?**
3. **What different Access Specifiers are in Ruby?**
4. **How they changes method visibilities ?**

Types of ARI

1. **Single Table Inheritanec (is_a relationship)**
2. **Multiple Table Inheritanec (has_a relationship)(has_one/has_many)**

Implementation step by step

Create new Rails App

```
rails _2.3.8_ ari -d mysql
    create
    create  app/controllers
    create  app/helpers
    create  app/models
    .     .     .     .
```

1. Single Table Inheritance (is_a relationship)

Generate Product migration

```
ruby script/generate model product type:string title:string color:string
    exists  app/models/
    exists  test/unit/
    exists  test/fixtures/
    create  app/models/product.rb
    create  test/unit/product_test.rb
    create  test/fixtures/products.yml
    create  db/migrate
    create  db/migrate/20101207145844_create_products.rb
```

Migrate Database

```
rake db:migrate

(in /home/sandip/ari)
==  CreateProducts: migrating============================================
-- create_table(:products)
   -> 0.0018s
==  CreateProducts: migrated (0.0019s) ===================================
```

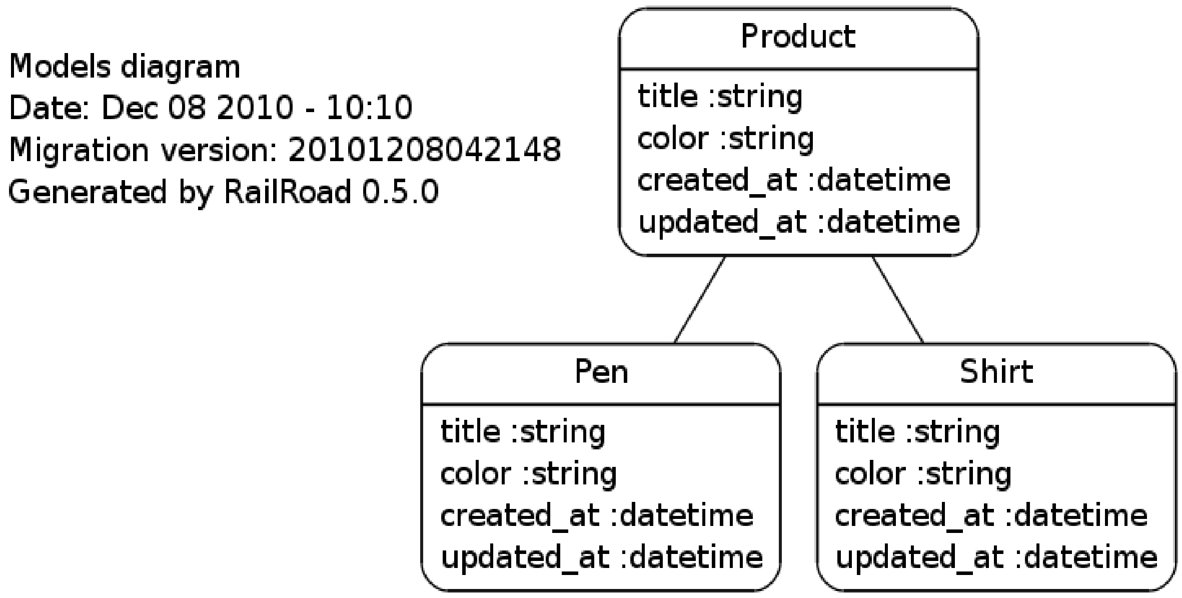
Ruby Console

```
script/console

Loading development environment (Rails 2.3.8)
>> Product.all
=> []
>> Product
=> Product(id: integer, type: string, title: string, color: string, created_at: datetime,
updated_at: datetime)
```

Models

```
vi app/models/pen.rb
cat app/models/pen.rb

class Pen < Product

end

vi app/models/shirt.rb
cat app/models/shirt.rb

class Shirt < Product

end
```

Active record Inheritance



Create initial Data

vi db/seeds.rb
cat db/seeds.rb

```
# This file should contain all the record creation needed to seed the database with its default values.
# The data can then be loaded with the rake db:seed (or created alongside the db with db:setup).
#
# Examples:
#
#   cities = City.create([{ :name => 'Chicago' }, { :name => 'Copenhagen' }])
#   Major.create(:name => 'Daley', :city => cities.first)

Pen.create(:title => 'ball pen', :color => 'red')
Pen.create(:title => 'ink pen', :color => 'blue')

Shirt.create(:title => 'Formal', :color => 'White')
Shirt.create(:title => 'T-shirt', :color => 'Blue')
```

Load Data

```
rake db:seed
(in /home/sandip/ari)
```

Database Record View

```
script/dbconsole
```

```
mysql> select * from products;

+----+-------+----------+-------+---------------------+---------------------+
| id | type  | title    | color | created_at          | updated_at          |
+----+-------+----------+-------+---------------------+---------------------+
|  1 | Pen   | ball pen | red   | 2010-12-07 15:08:46 | 2010-12-07 15:08:46 |
|  2 | Pen   | ink pen  | blue  | 2010-12-07 15:08:46 | 2010-12-07 15:08:46 |
|  3 | Shirt | Formal   | White | 2010-12-07 15:08:46 | 2010-12-07 15:08:46 |
|  4 | Shirt | T-shirt  | Blue  | 2010-12-07 15:08:46 | 2010-12-07 15:08:46 |
+----+-------+----------+-------+---------------------+---------------------+
4 rows in set (0.00 sec)
```

Ruby Console

```
script/console

Loading development environment (Rails 2.3.8)
>> Product.all.collect(&:to_s)
=> ["#<Pen:0xa5907a4>", "#<Pen:0xa58ff70>", "#<Shirt:0xa58ae6c>",
"#<Shirt:0xa58ad2c>"]

>> Pen.all.collect(&:to_s)
=> ["#<Pen:0xa57c5d8>", "#<Pen:0xa57c2b8>"]

>> Shirt.all.collect(&:to_s)
=> ["#<Shirt:0xa57727c>", "#<Shirt:0xa577100>"]
```

Rails Log View (Observe mySQL Queries)

```
Product Load (0.1ms)   SELECT * FROM `products`
Product Columns (0.9ms)   SHOW FIELDS FROM `products`

Pen Columns (0.9ms)   SHOW FIELDS FROM `products`
Pen Load (0.3ms)   SELECT * FROM `products` WHERE ( (`products`.`type` = 'Pen' ) )

Shirt Columns (0.8ms)   SHOW FIELDS FROM `products`
Shirt Load (0.3ms)   SELECT * FROM `products` WHERE ( (`products`.`type` = 'Shirt' ) )
```

Again On Console (Observe Class Hierarchy)

```
script/console
Loading development environment (Rails 2.3.8)

>> Product.superclass
=> ActiveRecord::Base

>> Pen.superclass
=> Product(id: integer, type: string, title: string, color: string, created_at: datetime,
updated_at: datetime)

>> Shirt.superclass
=> Product(id: integer, type: string, title: string, color: string, created_at: datetime,
updated_at: datetime)

>> Product.superclass.superclass
=> Object
```

Observe Public Method

```
cat app/models/product.rb

class Product < ActiveRecord::Base

  validates_uniqueness_of :title

  def to_param
    self.title
  end

end
```

On Console

```
script/console
Loading development environment (Rails 2.3.8)

>> p = Product.first
=> #<Pen id: 1, type: "Pen", title: "ball pen", color: "red", created_at: "2010-12-07
15:08:46", updated_at: "2010-12-07 15:08:46">

>> p.to_param
=> "ball pen"
```

Observe Private and Protected Methods

cat app/models/product.rb

```
class Product < ActiveRecord::Base

  validates_uniqueness_of :title
  def to_param
    self.title
  end

  protected
  def buy
  end

  private
  # private methods can be called in objects only
  # can not be invoked with instance
  # not even self.identity
  # private methods in ruby accessible to childs you can't completely hide a method(similar
to protected in java)
  def identity
    puts self.class
  end
end
```

Ruby Console

```
script/console
Loading development environment (Rails 2.3.8)

>> p = Product.first
=> #<Pen id: 1, type: "Pen", title: "ball pen", color: "red", created_at: "2010-12-07
15:08:46", updated_at: "2010-12-07 15:08:46">

>> p.buy
NoMethodError: protected method `buy' called for #<Pen:0xb1878c8>
```

Lets access private method inside class.

```
cat app/models/pen.rb

class Pen < Product

  def buy
    identity
    "Added to cart"
  end
end
```

On Console (Observer private methods can be accessed inside class)

```
script/console
Loading development environment (Rails 2.3.8)

>> p = Pen.first
=> #<Pen id: 1, type: "Pen", title: "ball pen", color: "red", created_at: "2010-12-07
15:08:46", updated_at: "2010-12-07 15:08:46">

>> p.buy
Pen
```

```
=> "Added to cart"
```

See the differenec between private and protected

So, from within an object "a1" (an instance of Class A), you can call private methods only for instance of "a1" (self). And you can not call private methods of object "a2" (that also is of class A) - they are private to a2. But you can call protected methods of object "a2" since objects a1 and a2 are both of class A.

gives following example - implementing an operator that compares one internal variable with variable from another class (for purposes of comparing the objects):

```
def <=>(other)
  self.identity <=> other.identity
end
```

2. Multiple Table Inheritance (Polymorphic has_a relationship)

Generate migration for Purchase

```
ruby script/generate model purchase resource_id:integer resource_type:string user_id:integer quantity:integer
```

Migrate

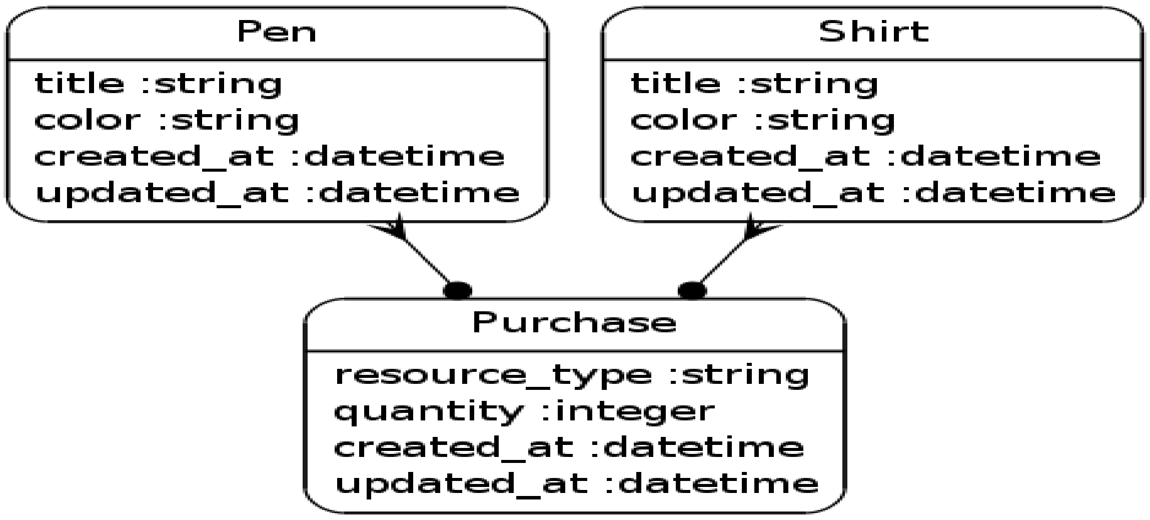
```
rake db:migrate
(in /home/sandip/ari)
==  CreatePurchases: migrating=========================================
-- create_table(:purchases)
   -> 0.2444s
==  CreatePurchases: migrated (0.2445s) ================================
```

Add polymorphic associations in Purchase and Product Model

```
cat app/models/purchase.rb

class Purchase < ActiveRecord::Base

  belongs_to :resource, :polymorphic => true
end
```

```
cat app/models/product.rb

class Product < ActiveRecord::Base

  has_many :purchases, :as => :resource, :dependent => :destroy
  validates_uniqueness_of :title

  def to_param
```

```
    self.title
  end

  protected
  def buy
  end

  private
  def identity
    puts self.class
  end
end
```



On Console

```
script/console
Loading development environment (Rails 2.3.8)

p = Pen.first
=> #<Pen id: 1, type: "Pen", title: "ball pen", color: "red", created_at: "2010-12-07
15:08:46", updated_at: "2010-12-07 15:08:46">

>> p.purchases
=> []

>> p.purchases.create(:quantity => 2)
=> #<Purchase id: 1, resource_id: 1, resource_type: "Product", user_id: nil, quantity:
2, created_at: "2010-12-08 13:47:32", updated_at: "2010-12-08 13:47:32">

>> p.purchases
=> [#<Purchase id: 1, resource_id: 1, resource_type: "Product", user_id: nil, quantity:
2, created_at: "2010-12-08 13:47:32", updated_at: "2010-12-08 13:47:32">]
```

```
>> Purchase.first.resource
=> #<Pen id: 1, type: "Pen", title: "ball pen", color: "red", created_at: "2010-12-07
15:08:46", updated_at: "2010-12-07 15:08:46">
```

End Of Session !!!

??????