



RailsConf '07 Tutorial

# Agenda

- Current Status
- Part 1: Architecture and Default Behavior
- Part 2: Streamlined Columns
- Part 3: Custom UI
- Part 4: Other Declarative Options
- Upcoming Features
- Contributing

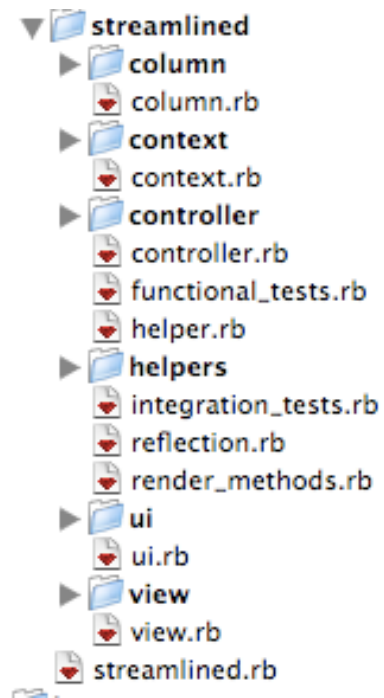
# Current Status

- Release 1.0
- Use in several commercial applications
- Community is small but growing

# Architecture

- Changed from Generator to Plugin with release 0.0.7
- `acts_as_streamlined` in controllers
- Controller now provides behavior for a single model

# Modular Infrastructure



# Integrates Into...

- ActionController
- Helpers
- Rendering
- Routing

# View Layer

- Provides default views for
  - list
  - show
  - edit
  - create
- Views are can be overwritten
  - Globally
  - per-Model

# Default List Behavior

- Paginated table
- All columns and relationships
- Pagination, sorting and filtering all handled via XHR calls
- Show, Edit and Delete buttons per row
- Create, Export (XML, CSV) for filter-set

# Default Edit/Create

- Non-Ajax Edit/Create screens
- Can assign related items during Edit OR Create

# Default Delete

- Uses JavaScript alert to confirm

# Declarative Changes

- Per-model declarative changes are made in app/streamlined folder
- One class per model
- Team class -> TeamUI class

# Streamlined Columns

- Everything is a column:
  - ActiveRecord attributes
  - Associations
  - Non-attribute methods
  - Custom Additions

# Default List

- All Associations
- Any ActiveRecord Attributes not called:
  - :id, :\*\_at, :\*\_on, :lock\_version, :\*\_id

# Override in [Model]UI

- Use `user_columns` to specify:
  - included columns
  - order
  - per-column options

# Example

```
class TeamUI < Streamlined::UI  
  user_columns      :id,  
                    :name,  
                    :sport  
end
```

# Options

- Make a column a link
- Change column label/header
- Create an Overlib popup
- Mark it read-only or create-only

# Example

```
class TeamUI < Streamlined::UI

  user_columns    :id, {:read_only => true},
                  :name, {:link_to => {:action => 'edit'}},
                  :sport, {:popup => true}

end
```

# Enumeration Columns

- Column can contain value from another table
- For example, Team.sport can be one of:
  - ['Baseball', 'Basketball', 'Football']

# Example

```
class TeamUI < Streamlined::UI

  user_columns :id, {:read_only => true},
               :name, {:link_to => {:action => 'edit'}},
               :sport, {:popup => true,
                       :enumeration => Sport.SPORTS}

end

class Sport
  SPORTS = ['Baseball', 'Basketball', 'Football']
  # or
  def SPORTS
    Sport.find(:all).collect{|s| s.name}
  end
end
```

# Options for Associations

- Change Show view
- Change Edit view

# Example

```
class TeamUI < Streamlined::UI

  user_columns :id, {:read_only => true},
               :name, {:link_to => {:action => 'edit'}},
               :sport, {:popup => true},
               :coach, {
                 :show_view => {:name,
                               {:fields => [:first_name, :last_name],
                               :separator => " "}},
                 },
               :edit_view => {:select}}

end
```

# Available Show Views

Name	Use	Available
Link	Link to related table	All
Count	Shows number of related items	1-to-n
List	Display data from related models	1-to-n
Sum	Calculate sum of field on related items	1-to-n
Average	Calculate average of field on related items	1-to-n
Name	Display data from related item	n-to-1
Enumerable	Treat association as an enumerable	n-to-1
Graph	Display sparklines graph of related items	1-to-n

# Available Edit Views

Name	Use	Available
Inset Table	Master-detail view	Any
Membership	Allows user to select items	1-to-n
Polymorphic Membership	Allows user to select related items of polymorphic relationship	1-to-n
Window	Shows related items in table in popup window	1-to-n
Select	Select related item from dropdown	n-to-1
Polymorphic Select	Select related polymorphic item from dropdown	n-to-1
Enumerable Select	Treat association as an enumerable	n-to-1
Filter Select	Like membership, but uses Ajax to allow filtering of possible matches	1-to-n

# Association Components

- The available views are the stock set
- Can be modified
- New ones can be added easily

# Columns for Edit/Show/ Create

- Can redefine column list per CRUD context
  - `show_columns`
  - `edit_columns`
  - `list_columns`
  - `create_columns`

# Using non-AR Methods

- Simply need to specify the method name in `user_columns`
- Can also use the Additions module to keep view-related clutter out of the AR model

# Example

```
module TeamAdditions
  def mascot_picture
    "<img src='/#{self.name}/mascot.jpg'>"
  end
end

Team.class_eval {include TeamAdditions}

class TeamUI < Streamlined::UI

  user_columns :id, {:read_only => true},
               :name, {:link_to => {:action => 'edit'}},
               :sport,
               :mascot_picture, {:human_name => 'Mascot'}
end
```

# Customizing the UI

- Default views maintained in `app/vendor/plugins/streamlined/views`
- Can override globally or per-model

# Global Override

- Replace any view file by name in app/  
streamlined/views

# Per-Model Override

- Replace any view by name in `app/views/`  
`[model]/`

# Overriding Controller Behavior

- Controller behavior is mixed in through `acts_as_streamlined`
- Redefine methods to change their behavior
- Add `around_filters` to wrap the behavior

# Other Options

- Choosing link styles
- Controlling top and side menus
- Unassigned\_if\_allowed
- Overriding model name

# Choosing Link Styles

- Can override link styles by including appropriate Helper
- Default is `Streamlined::Helpers::LinkHelper`
- Ajax is `Streamlined::Helpers::AjaxLinkHelper`

# Top and Side Menus

- Defined in Helpers
- `streamlined_top_menus`,  
`streamlined_side_menus`

# Example

```
module ApplicationHelper
  def streamlined_top_menus
    [
      ["Teams", { :controller => "teams", :action => "index" }],
      ["Coaches", { :controller => "coaches", :action => "index" }]
    ]
  end
end

module TeamHelper
  def streamlined_side_menus
    [
      ["New", { :action => "create" }],
      ["List", { :action => "index" }]
    ]
  end
end
```

# Unassigned\_if\_allowed

- By default, Streamlined looks at validations on models to determine if a relationship is allowed to be null
- If so, “unassigned” will be placed as first selection choice
- Otherwise, it will not be in the list

# Overriding Model Name

- By default, model is determined from controller name
- Can override using `streamlined_model`

# Example

```
class TeamController < ApplicationController
  layout 'streamlined'
  acts_as_streamlined
  streamlined_model 'SportTeam'
end
```

# Upcoming Features

- Editable associations: edit associated models in-line
- Tabbed Views
- Per-user column preferences
- Smart folders/searching
- Persistent Filters
- More

# Contributing

- We want your help
- Blog: [streamlinedframework.org](http://streamlinedframework.org)
- Links to Trac, Wiki, Discussion Group
- Please come help out!