DREW BISSET

WEB APPLICATION DEVELOPER

andrew.bisset@gmail.com
https://github.com/drewdev
(617) - 863-0080

Profile

Passionate for elegant and lightweight web software solutions which model the problem domain acutely and future proof its interfaces over the implementation.

Skills

Modern Web Development

MVC, Active Record ORM, Active Support Concerns, Strong Parameters ~ I love Rails!
Method Missing, Refinements, Blocks, Attributes ~ I love Ruby!
Interactive Rebase, Cherry Pick, Stash ~ I love Git!
Registers, Macros, :map, Tim Pope ~ I love Vim!

TDD/BDD

Proponent for designing from the outside-in using testing methodologies that help break down complexity into logical units and support self-documenting feature development.

Agile

"Software isn't hard, people are." Let's scaffold the solution with regular - and, when possible, real - human interaction. Frequent standups, pair-programming, honest and open-minded communcation, making small things, collaboratively - these behaviors predict agility.

Technical

Haml	Postgres	Sinatra	MVC
CoffeeScript	Mongo	Rails Engines	Strategy Pattern
Jbuilder	Redis	Padrino	Actor Model

Experience

Berklee Online

2012-present

Web Software Developer

Work with business teams to implement solutions using Ruby and Ruby on Rails. Manage work through user stories defined in Pivotal Tracker and implemented using TDD. Deploy often to AWS/Heroku using Jenkins for continuous integration and Git for distributed source control.

Delivered a green field rewrite of Berklee's front end catalog web site using Ruby on Rails. Deliverables included:

- Test suite with Turnip integration tests and unit tests in RSpec
- ActiveSupport Concerns to encpasulate behavior used across various models in similar ways
- An ActiveRecord driven schema supported by a legacy namespace holding integration points to data from collaborating systems
- Deployment to a new domain name including migration of SEO page rankings
- A mechanism for handling cross-domain authentication with the legacy domain name where the provider and many other authenticated systems still live

Prototyping a new Learning Management System using many modern tools and frameworks including:

- Ruby on Rails for the backend API, MongoDB for the persistence layer, and AngularJS for the front end single page application experience.
- Many of the mainstay Node.js build tools for supporting AngularJS development including Yeoman for scaffolding, Grunt for tasks, and Bower for asset package management.
- End-to-End testing with Protractor and integration testing with Jasmine

In addition, research is also invested into how a third party rails-based service like Canvas could be leveraged and extended over building part or all of the LMS system in house

Commonwealth Financial Network

2007-2012

Web Application Developer

Delivered web-based solutions using ASP.NET and SQL Server for use by both internal staff and financial advisor clients and their respective staff. These included:

- Data Reporting using SSRS, Crystal, and homegrown reporting solutions
- Financial forms and compliance workflows using ASP.NET Web Forms
- Extending Microsoft CRM with custom plugins along with external dashboards/widgets consuming MSCRM web resources

Delivered several from scratch solutions from requirements gathering to production release. These included a single-page application which centralized, annotated, and aided in the discovery of the firm's many marketing materials for use by clients. Implementation included:

- Working regularly with business stakeholders to help flesh out certain ideas and interactions along with prioritizing work items against deadlines
- Implementing a data repository layer to isolate the SharePoint data provider away from the core business objects consumed by the client
- Applying an MVC design to the custom javascript modules written to support the SPA behaviors

In addition to a bi-lateral sync enginge that married the data between two independent CRM systems used by different departments within the busines. The system was designed to:

- Minimize the footprint on the systems being synchronized by using execution plans to tune stored procedure code and setup an isolated staging database environment
- Store snapshots of each sync execution allowing for identification and debugging of data anamolies
- Use an elastic scope of changes to synchronize being driven by the last successful iteration. This allowed for sync downtime, planned or otherwise, with no impact of overlooking unsynched changes to data

Education

New England Conservatory - Boston, Massachusetts

Bachelor of Arts in Music Composition
Concnetration in Music-In-Education

Coursera - Online

SaaS Class Certificate of Completion
http://saas-class.org

Interests

Producing and DJing	Rock Climbing/Bouldering	Coffee Roasting/Brewing
Guitar, Piano, Drums	Cycling	Mixology
Classical Composition	Skiing	Cricket Harvesting

Drew Bisset — andrew.bisset@gmail.com — (617) - 863-0080