

FRAMEWORKS

COLABORADORES:

- .NET - Rui Marques
- Revel - Florêncio Alexandre
- Beego - Florêncio Alexandre
- Ruby on Rails - Daniel Casimiro
- Laravel - João Canoso
- Slim - João Canoso
- Sails.JS - Rui Barcelos
- Spring - Diogo Mendes
- Dropwizard - Diogo Mendes
- Spark - Luís Nunes
- Django - David Bernardo
- Catalyst - Luís Marques

COLAGEM:

- Diogo Mendes



.NET DESCRIÇÃO

What's new in .NET Framework 4.5?

Windows Presentation Foundation

Built-in Ribbon controls ★
 Databinding improvements
 Ability to add breakpoints to databindings
 Data source change aware views (Live Shaping) ★
 Validation improvements A
 Improved legacy UI integration
 Dispatcher improvements A
 Speed-up of large data sets P

Windows 8 support ★

- Support for Windows Runtime (WinRT)
- .NET Profile for Metro-style apps
- Improved support for sharing DLLs between .NET profiles

ASP.NET

- ASP.NET Web API - a new framework for REST endpoints
- Support for implementing WebSocket receivers
- Built-in JavaScript + CSS combining and minification
- Support for Recipes: intelligent codegen ★
- Asynchronous pipeline support A (Response, Request, HttpHandlers)
- Performance improvements:** Multicore JIT, 35 % faster startup, memory optimizations, assembly sharing between sites, pre-fetch support P

ASP.NET MVC 4

- Async controllers A
- Built-in mobile templates + jQuery.Mobile support
- Alternate views (e.g. print version, mobile site)

ASP.NET Web Pages 2

- New site templates
- Versatile validation support
- Support for OAuth and OpenID ★
- Built-in map embedding tools; supports Google, Bing and others
- Did you know?** ASP.NET Web Pages is yet another way to work the web besides Web Forms and MVC. WP sites use Razor and are typically developed with WebMatrix.

Web Forms

- Strongly typed data binding
- MVC-like support for Models ★
- HTML encoded binding expressions
- HTML5 support** ★ Control support for new semantic elements
- Multi-file support for FileUpload control
- Validator and UpdatePanel now support new HTML5 elements

Tooling

- Markup editors improved (a lot)
- IIS Express used by default
- Page Inspector ★
- New templates and snippets

Windows Communication Foundation

- New channels (UDP multicast, WebSockets, ...) ★
- Support for contract-first development ★
- Asynchronous operations A
- Streaming improvements
- Simplified configuration, VS config validation
- Multiple auth modes for HTTP endpoints
- New, simple HttpClient class

Windows Workflow Foundation

- C# Expressions
- State machine workflows are back! ★
- Workflow versioning ★
- Code-first activity design
- Faster execution P
- Designer usability improvements

Managed Extensibility Framework 2.0

- Support for generic parts ★
- Debugging improvements
- Support for explicit and convention-based bindings between objects ★
- Support for binding POCOs: no more attribute requirements

ADO.NET

- Sparse columns support improved (SQL Server)
- Passwords are now stored encrypted
- Asynchronous operations A

SQL Express LocalDB

- New light version of SQL Express for developer use. Supported in .NET 4.5, separate patch for 4.0 is coming.

SQL Server 2012 ("Denali") Support

- High Availability support on connection string level
- Fast failover across multiple subnets
- Support for new spatial data types (polygons, arcs etc.) ★

Entity Framework 4.5

- Enumeration support
- Migrations for schema changes ★
- Designer improvements

- Spatial data type support ★
- Table-valued function support ★
- Multi-result sproc support

- Multiple diagrams per model
- Code-first support ★
- Auto-compiled LINQ queries P

Base Class Library

- Networking improvements (IPv6 enhanced, IDN, EAI etc.)
- Key interfaces (e.g. file IO) now support async A
- New ArraySegment and ReadOnlyDictionary classes
- Support for CLR objects over 2 GB in size
- Resource file management performance improved P
- Unicode improvements (v6, console support)
- Background JIT on multicore platforms P

Task Parallel Library A

- Task thread controls improved: Task.WaitAll/.WaitAny, various timeout primitives available
- TPL Dataflow: Tools for parallel data flow processing ★

C# 5.0

- Support for async programming: async and await keywords A
- Methods can access call site info as parameters (CallerInfo)

Visual Basic 11

- Iterator implementations (Yield)
- Async and Await equal to C# A
- "Global" keyword for namespace referencing
- Call Hierarchy view available
- CallerInfo attributes (as in C#)

F# 3.0

- Type providers ★
- Query expressions (LINQ) ★
- Auto-implemented properties

Visual C++ 11

- C++11 standard support improved
- Auto-vectorization and A parallelization of loops P
- GPU-driven processing (C++ AMP)
- Intellisense for C++/CLI

Legend:

- A = Asynchrony support
- P = Performance improvement
- ★ = Significant new feature

.NET - VANTAGENS

- Suporte para mais de 20 linguagens
- Grande quantidade de elementos gráficos disponíveis
- IDE poderoso e versátil.
- Enorme comunidade de suporte
- Compatível com um grande numero de motores de BD
- Como as linguagens são de alto nível e compiladas, torna-se mais fácil de detetar erros
- Já é possível utilizar .NET em outras plataformas não Microsoft
- Existe uma versão totalmente gratuita do IDE
- Validações em controlos simples de executar através das propriedades dos mesmos
- É uma framework bastante madura

.NET - DESVANTAGENS

- A .Net limita a escolha do hardware, sistema operativo e middleware
- Apesar do grande numero de bibliotecas existentes, poucas são aquelas que são gratuitas e open source
- O IDE apesar de poderoso, torna-se complexo de trabalhar
- Como o HTML é auto gerado pela aplicação aquando dos requests dos clientes, o mesmo não fornece grande controlo sobre "que" HTML é gerado.
- Otimizada para trabalhar com MSSQL, sendo que este não é propriamente "simpático" de configurar e trabalhar.
- Custos mais elevados de desenvolvimento e manutenção em relação a outras frameworks
- Sem o codigo auto gerado pela aplicação, o desenvolvimento é moroso.
- O código ao ser compilado, obriga que qualquer alteração obrigue a uma recompilação do mesmo
- O IIS apesar de bastante flexível, é difícil de implementar e configurar.



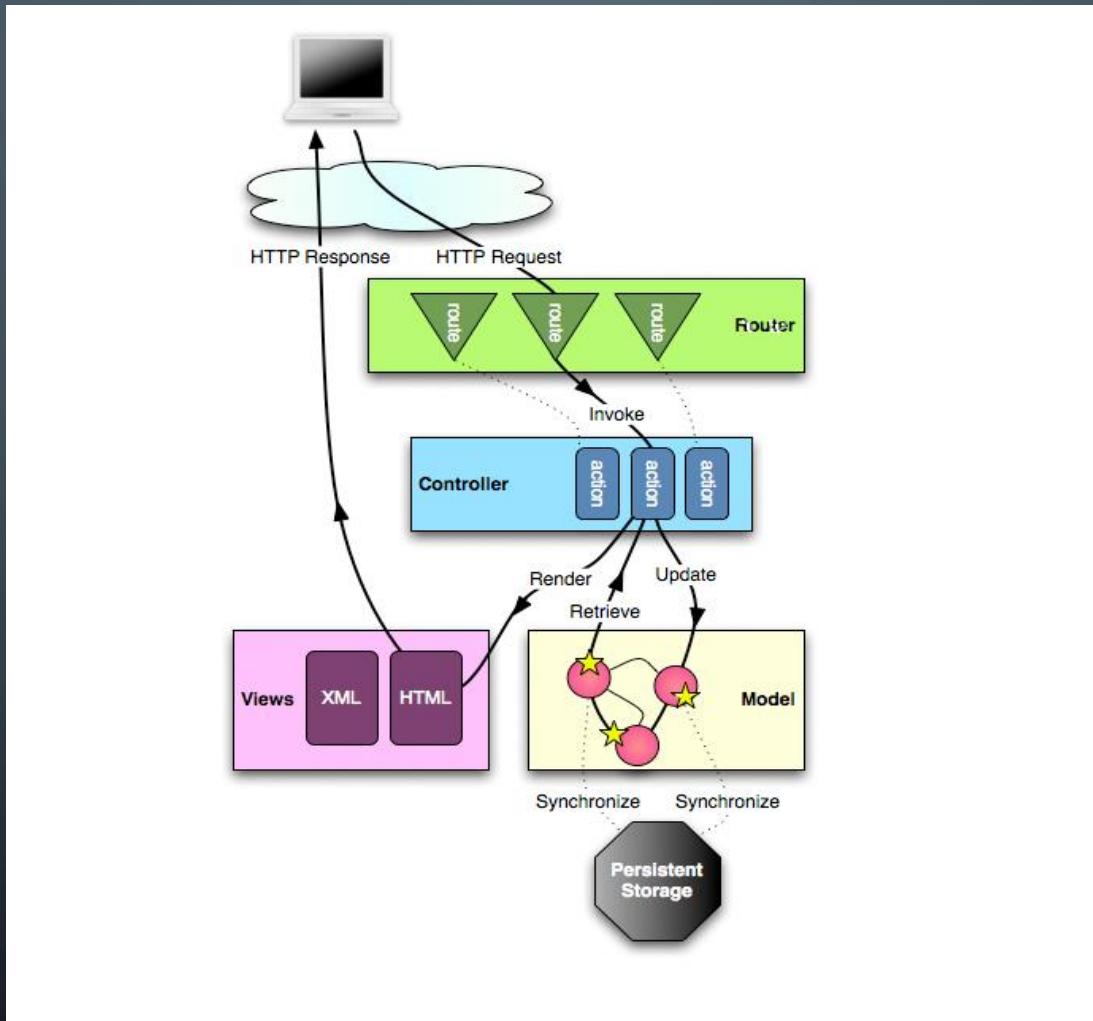
Revel

A high-productivity web framework for the Go language.

REVEL - DESCRIÇÃO

- Utiliza GoLang na sua base.
- Tenta tornar mais fácil a criação de aplicações Web usando o padrão Model-View-Controller (MVC), dependendo de convenções que exigem uma determinada estrutura numa aplicação.
- Configuração leve e permite ciclos de desenvolvimento muito rápidos

REVEL - ARQUITECTURA



REVEL - VANTAGENS

- Interface intuitiva
- Ajuda com dicas a resolver problemas e a implementar código
- Alta performance
- Fácil de aprender a usar
- Utiliza o standard MVC, um conceito que já foi provado que funciona
- Instalação e configuração muito leve
- Desenhado para ciclos de desenvolvimento rápidos
- Utiliza Filters e Interceptors
- Revel Tool - uma ferramenta de suporte a desenvolvimento

REVEL - DESVANTAGENS

- Não suporta MongoDB
- Falta de suporte para várias ferramentas de suporte standard
- Não suporta muito bem Jobs

B E E G O

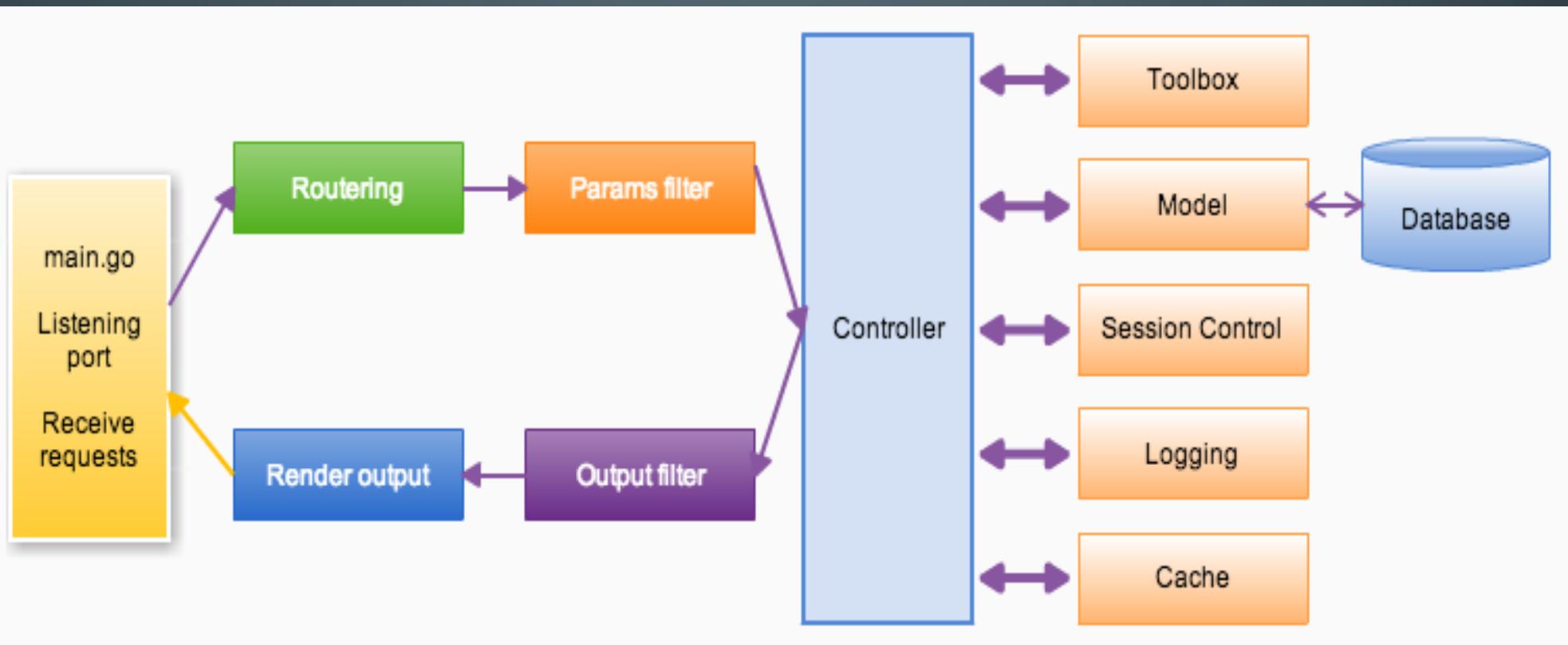


beego

BEEGO - DESCRIÇÃO

- O Beego é uma Framework HTTP para o rápido desenvolvimento de aplicações Go.
- Este pode ser usado para desenvolver APIs, aplicações web e serviços back-end rapidamente.
- É uma estrutura RESTful.
- É inspirado por Tornado, Sinatra e Flask e integra características específicas de Go , tal como o sistema de interfaces e utilização de structs

BEEGO - ARQUITECTURA



BEEGO - VANTAGENS

- Robusta
- Rápida
- Fácil de se utilizar
- Intuitiva
- Biblioteca bastante completa
- Boa documentação
- Suporta SQL/ORM
- Bee Tool - uma ferramenta de suporte a desenvolvimento
- Suporta todas as funcionalidades que Go tem para oferecer
- Utiliza Swagger para documentação

BEEGO - DESVANTAGENS

- Por vezes quando a build falha, o sistema não notifica
- Não suporta MongoDB



RUBY ON RAILS - DESCRIÇÃO

- Rails é um "meta-framework" (ou seja, um framework de frameworks), baseado em Ruby. Este "meta-framework" é composto pelos seguintes frameworks:
 - Active Record: É uma camada de mapeamento objeto-relacional, responsável pela interoperabilidade entre a aplicação e o base de dados e pela abstração dos dados.
 - Active Resource: Providencia uma interface para comunicação entre aplicações Rails.
 - Action Pack: É Composto pelo Action View (gerador de vistas) e o Action Controller (controlo do fluxo de negócio).
 - Action Mailer: É um framework responsável pelo serviço de entrega e receção de e-mails.
 - Active Support: É uma coleção de várias classes úteis e extensões de bibliotecas padrões, que foram considerados úteis em aplicações do tipo Ruby on Rails.
 - Action Cable: Action Cable é uma framework para comunicações em tempo real por web sockets

RUBY ON RAILS - VANTAGENS

- Rails utiliza a arquitetura RESTful por defeito e apresenta um sistema de rotas avançado
- Tempo de desenvolvimento rápido (usando todos os recursos do Rails (uso dos geradores))
- Modo API introduzido com o Rails 5
- O Rails foi Desenhado tendo em conta a metodologia de desenvolvimento AGILE
- O Rails é uma framework madura com ferramentas de teste integradas
- Não é necessário reinventar a roda a framework Rails faz muita coisa pelo programador, é possível criar uma aplicação com funcionalidades básicas como login e registo de utilizadores com papéis, gestão de utilizadores e um layout bem desenhado sem escrever uma única linha de código.
- Funcionalidade básicas podem ser geradas pelo Rails em vez de terem que ser programadas
- Rails cobre tanto front-end como back-end permitindo construir uma aplicação web completa sem ser necessário utilizar outra ferramenta.
- Rails trata da maioria das questões de segurança como sql injection, cross-site scripting, session hijacking e outros
- Segue com algum rigor o modelo MVC (torna-se difícil aligeirar o modelo ou utilizar outro modelo)

RUBY ON RAILS - DESVANTAGENS

- Ruby é uma linguagem difícil de aprender VS Javascript (Node.JS) e tem a curva de aprendizagem íngreme
- Rails é Ineficiente, difícil de escalar e é complexo compreender o ecossistema do Rails
- Ruby perdeu popularidade nos últimos anos
- O sistema de processamento HTTP do Rails é baseado numa filosofia de ser liberal naquilo em que se aceita. Por isso os programadores acabam inadvertidamente a enviar pedidos à API que não são documentados ou nem estão de acordo com o RFC's, no entanto o middleware do Rails limpa os pedidos e lida com eles muito bem como se fossem pedidos válidos.
- O Rails abstrai o programador de algumas tarefas (exemplo: torna transparente para o utilizador interagir com a base de dados), isto é esconde muita coisa que o programador teria que conhecer.
- O Ruby foi desenhado de forma a manter os programadores contentes (sendo assim existem múltiplas formas de escrever código que faz a mesma coisa, e muitos relaxamentos na forma de escrever código)
- Twitter abandonou completamente a utilização do Ruby e o Rails em 2011 devido a preocupações com a escalabilidade e performance

RUBY ON RAILS

- Github
- Hulu
- Indiegogo
- Kickstarter
- Scribd
- SlideShare
- Strava
- ThemeForest
- Twitch
- SoundCloud



- SERVIÇOS CONSTRUIDOS COM RAILS

Scribd.



STRAVA™



hulu



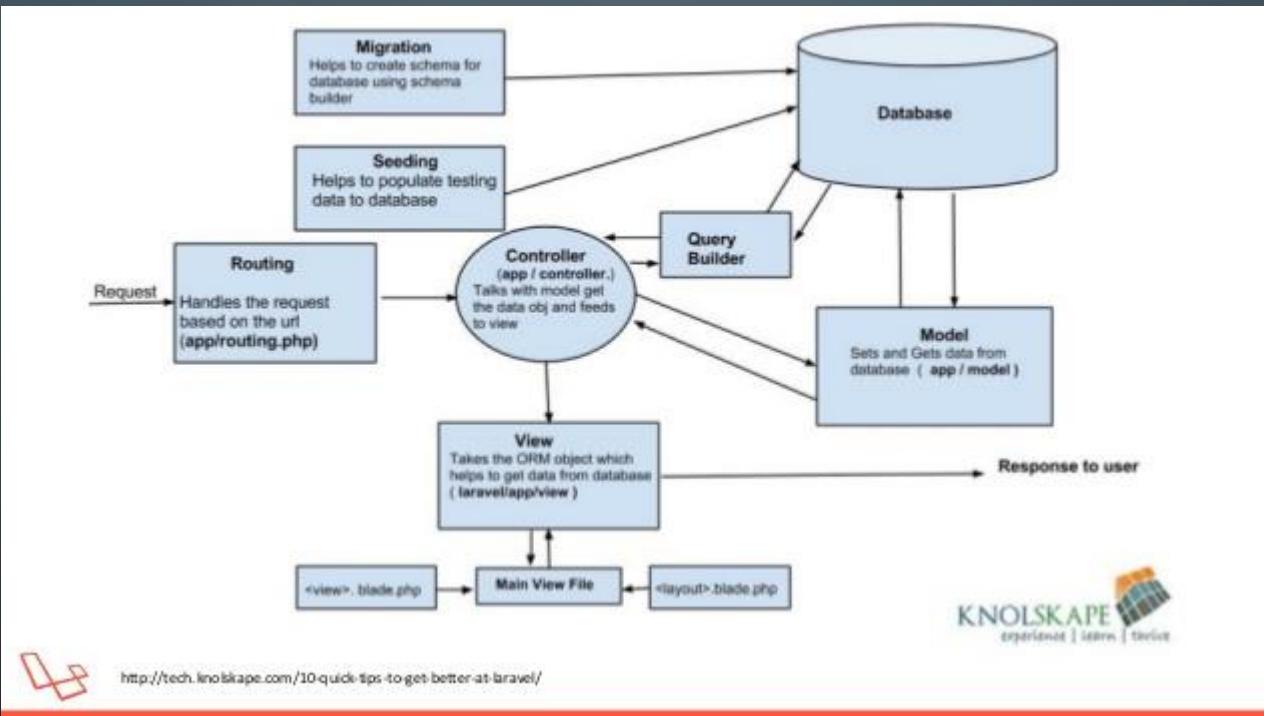


LARAVEL
PHP FRAMEWORK FOR WEB ARTISANS

LARAVEL - DESCRIÇÃO

- Laravel é uma framework PHP open-source utilizada para o desenvolvimento de aplicações web com o padrão MVC.
- Possui uma sintaxe simples e concisa, tento como tema “Love beautiful code? We do too”.
- Tem uma documentação ótima, com imensos videos.
- Possui um template engine (Blade) que facilita a criação de autenticação, sessões, caching e RESTful routing.
- Utiliza o Artisan para efetuar diversas tarefas como migrações de BD.

LARAVEL - ARQUITETURA



LARAVEL - VANTAGENS

- É open-source.
- Tem uma ótima documentação e uma comunidade ativa.
- É fácil de aprender.
- É escalável.
- Utiliza o modelo MVC .
- Possui uma CLI (artisan) que possui ferramentas para administrar a app.
- Utiliza um template engine (Blade) super rápido que compila o php e faz cache dos resultados.
- É implementada com o Composer (gestor de dependências do PHP).
- Permite routing reverso.

LARAVEL - DESVANTAGENS

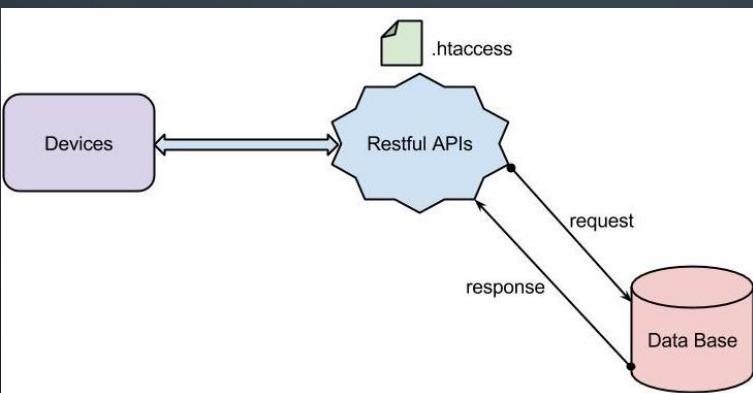
- O desenvolvimento é mais lento quando comparado com Ruby on Rails.
- É pesada quando comparada com outras frameworks.
- Executa muitas querys na base de dados.

Slim

a micro framework for PHP

SLIM - DESCRIÇÃO

- Slim é uma micro framework que facilita a criação de simples e poderosas API's.
- Esta framework é utilizada principalmente no desenvolvimento de RESTful API's e serviços.
- Dispõe de diversas funcionalidades como routing de URL's, caching HTTP, cookies e sessões seguras, log's e debugging.



SLIM - VANTAGENS

- Implementa RESTful cumprindo os padrões da industria.
- É simples e rápida, precisando de poucos recursos.
- A documentação está organizada e detalhada.
- Contem classes para gerir pedidos, respostas, cookies, caching, log's, etc.

Exemplo de implementação

```
<?php
$app = new \Slim\Slim();
$app->get('/hello/:name', function ($name) {
    echo "Hello, $name";
});
$app->run();
```

SLIM - DESVANTAGENS

- A falta de exemplos pode dificultar a aprendizagem.
- Pode ser uma framework simples de mais para implementar em projetos mais arrojados.

sails



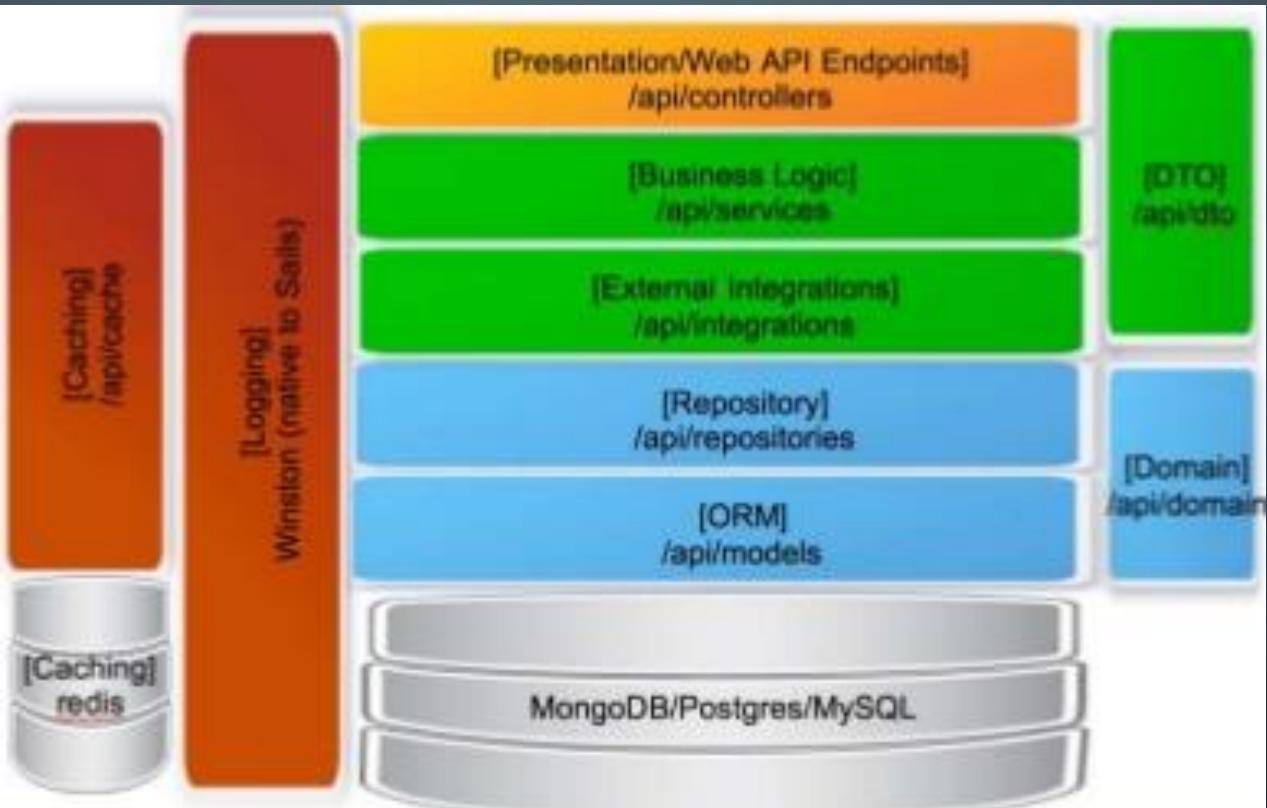
The web framework of your dreams



SAILS.JS - DESCRIÇÃO

- Serve para construir aplicações práticas e prontas para produção em Node.js em semanas e não meses.
- É uma das estruturas MVC mais popular para Node.js.
- Projetada para emular o padrão MVC familiar de estruturas como o Ruby on Rails.
- Tem suporte para os requisitos de aplicativos modernos: APIs orientadas a dados com uma arquitetura escalonável orientada a serviços.

SAILS.JS - ARQUITETURA



Data de Pesquisa: 2 de março de 2017

URL: <https://pt.slideshare.net/ericnograles/come-sail-away-with-me-you-guys-nodejs-mvc-web-apis-using-sailsjs>

SAILS.JS - VANTAGENS

- É 100% JavaScript;
- Usa qualquer base de dados;
- Gera Automaticamente REST APIs;
- Suporta WebSockets sem código adicional;
- As políticas de segurança são declarativas e reutilizáveis;
- Front-end é agnóstico;
- Pipeline de ativos flexíveis;

SAILS.JS - DESVANTAGENS

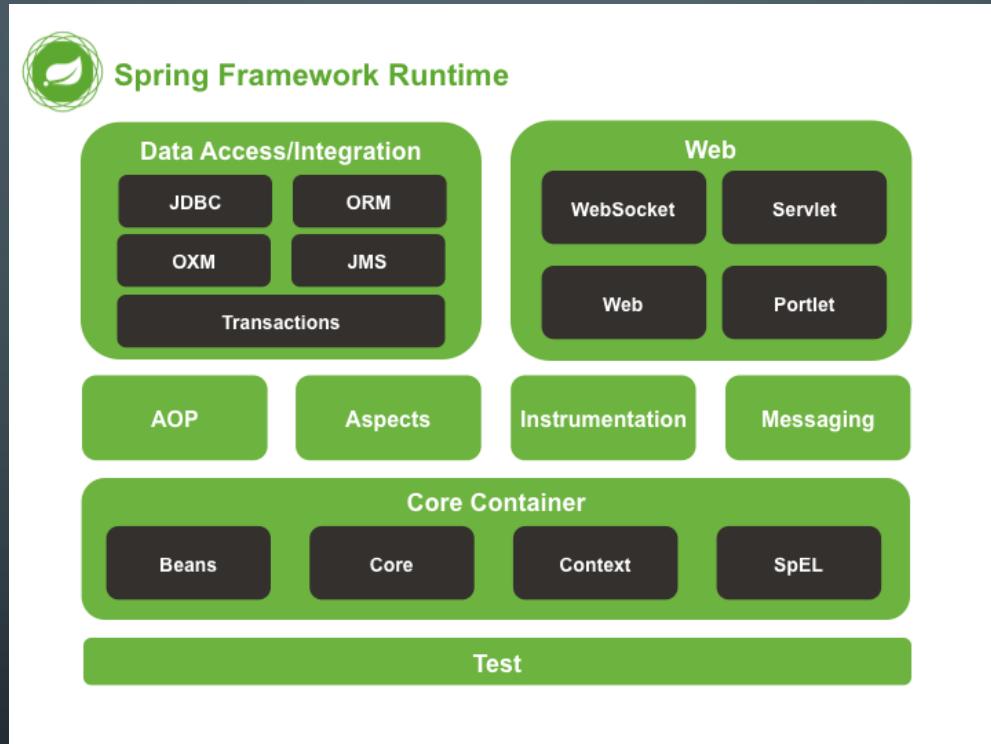
- Falta de documentação e de suporte.
- Demora demasiado tempo a iniciar e tem de ser reiniciado com frequência.
- Quando são feitas as questões no Github não são respondidas
- Não existe políticas de divulgação responsável documentada.
- Não existe informações sobre como deve de tratar as vulnerabilidades de segurança.
- Existe erros de digitação codificados nos argumentos do comando.



SPRING - DESCRIÇÃO

- O Spring Framework fornece um modelo abrangente de programação e configuração para aplicativos corporativos modernos baseados em Java, sem vínculos desnecessários com ambientes de implementação específicos centrando-se na lógica de negócios ao nível de aplicação.
- Tem suporte para funcionalidades como injeção de dependência, programação orientada aos aspetos, incluindo a gestão declarativa de transações da Spring, baseada em MVC e estrutura um serviço de Web RESTful
- Suporte básico para JDBC, JPA, JMS

SPRING - ARQUITETURA



SPRING - VANTAGENS

- Estruturada no modelo MVC.
- Suporte para tecnologias conhecidas como ORM (object-relational mapping), logging frameworks, JEE (java enterprise edition), JDK timers and Quartz.
- Facilmente testável.
- Flexibilidade nas relações de herança entre os objetos.
- Fácil tradução de exceções direcionadas pela tecnologia lançada, como JDBC (java database connector), Hibernate ou JDO (java data objects).
- Facilmente escalável.
- Grande comunidade de suporte, e tutoriais de aprendizagem na rede.
- Arquitetura modular.
- Framework completamente "view-agnostic", não necessita obrigatoriamente de usar JSP (java serve pages) para representar as views mas pode optar por Velocity ou XSLT.

SPRING - DESVANTAGENS

- Framework extremamente complexa, composta por mais de 2400 classes e 49 ferramentas;
- Curva de aprendizagem demasiado longa;
- Demasiado uso de código XML;
- Problemas de documentação para implementação de segurança no sistema;



MAKE FEATURES NOT WAR

DROPWIZARD

Make features not WAR

DROPWIZARD - DESCRIÇÃO

- Dropwizard é uma framework open-source em java para desenvolver API's restful de alto desempenho.
- Fornece um conjunto de bibliotecas incorporadas que consistem nos seguintes componentes:
 - Jetty – fornece um ficheiro .jar em vez de .war e inicia o seu próprio container em vez de evocar um servlet externo.
 - Jersey – equivalente ao JAX-RX do Spring para escrever web services restful.
 - Jackson – processa estruturas JASON.
 - Logging – através de Logback ou SLF4J.
 - Metrics – Fornece intuições sobre o que o código executa na produção.

DROPWIZARD - VANTAGENS

- Boa documentação
- Performance na manipulação de dados
- Facil estruturação
- Fornece informação acerca do tempo de resposta e de solicitação
- Cada API tem o seu próprio conteiner que permite o debug e a compilação do programa principal no próprio IDE evitando a recopilação do WAR



The logo consists of the word "spark" in a bold, black, sans-serif font. Behind the letter "p", there is a graphic element resembling a stylized sun or fire, composed of several sharp, radiating orange and yellow triangles.

spark

SPARK - DESCRIÇÃO

- O Spark é um uma micro framework aplicacional de código aberto escrito em Java.
- Não se deve confundir com o Apache Spark.
- Esta framework foi criada com o objetivo de ser fácil e rápido a criação de APIs.
- A biblioteca é leve e permite definir rotas e enviá-las para funções que responderão quando esses caminhos forem requeridos.
- Não segue o modelo model–view–controller (MVC)
- Centra-se na filosofia do Java 8 lambda.
- É ideal para RESTful API's.

SPARK - VANTAGENS

- **Velocidade:** apresenta um ranking muito alto na industria de benchmarks, onde não só o Java é testado mas muitas linguagens de programação e ferramentas.
- **Produtividade:** Um dos objetivos do Spark é tornar encaminhar os endpoints da API para os manipuladores.
- **Propósito de Construção:** Spark foi construído para fazer mais do que uma coisa bem feita – roteamento.
- **Pronto para a Cloud:** O Spark é uma excelente alternativa para outras framework pesadas fazendo-o perfeito para aplicações alojadas na cloud.

SPARK - CÓDIGO EXEMPLO

```
import static spark.Spark.*;  
  
public class HelloWorld {  
    public static void main(String[] args) {  
        get("/hello", (request, response) -> "Hello World");  
    }  
}
```

É possível verificar com este exemplo a simplicidade e facilidade de criar a API com o Spark

```
    class Poll(models.Model):
        question = models.CharField(max_length=200)
        pub_date = models.DateTimeField('date published')
        votes = models.IntegerField()
        def was_published_recently(self):
            return self.pub_date >= timezone.now() - datetime.timedelta(days=1)
        def __unicode__(self):
            return self.question
        @method_decorator(login_required)
        def detail(request, poll_id):
            p = get_object_or_404(Poll, pk=poll_id)
            try:
                selected_choice = p.choice_set.get(pk=request.POST['choice'])
            except (KeyError, Choice.DoesNotExist):
                messages.error(request, "You didn't select a choice.")
            else:
                selected_choice.votes += 1
                selected_choice.save()
            return HttpResponseRedirect(reverse('polls:results', args=(p.id,)))
        class Meta:
            ordering = ('-pub_date',)

    class Animal(models.Model):
        name = models.CharField(max_length=50)
        sound = models.CharField(max_length=50)
        def __unicode__(self):
            return self.name
        @method_decorator(login_required)
        def speak(request, animal_id):
            a = get_object_or_404(Animal, pk=animal_id)
            if request.method == 'POST':
                a.sound = request.POST['sound']
                a.save()
                return HttpResponseRedirect(reverse('polls:speak', args=(a.id,)))
            else:
                return render_to_response('polls/speak.html', {'animal': a})
        class Meta:
            ordering = ('name',)

    class Beat(models.Model):
        title = models.CharField(max_length=100)
        description = models.TextField()
        def __unicode__(self):
            return self.title
        @method_decorator(login_required)
        def feed(request, beat_id):
            b = get_object_or_404(Beat, pk=beat_id)
            if request.method == 'POST':
                b.description = request.POST['description']
                b.save()
                return HttpResponseRedirect(reverse('polls:feed', args=(b.id,)))
            else:
                return render_to_response('polls/feed.html', {'beat': b})
        class Meta:
            ordering = ('title',)

    class Crime(models.Model):
        beat = models.ForeignKey(Beat)
        date = models.DateTimeField()
        def __unicode__(self):
            return "%s %s" % (self.beat, self.date)
        @method_decorator(login_required)
        def report(request, beat_id):
            beat = get_object_or_404(Beat, pk=beat_id)
            if request.method == 'POST':
                c = Crime()
                c.beat = beat
                c.date = request.POST['date']
                c.save()
                return HttpResponseRedirect(reverse('polls:report', args=(c.id,)))
            else:
                return render_to_response('polls/report.html', {'beat': beat})
        class Meta:
            ordering = ('-date',)

    class Person(models.Model):
        first_name = models.CharField(max_length=50)
        last_name = models.CharField(max_length=50)
        sex = models.CharField(max_length=1, choices=((M, 'Male'), (F, 'Female')))
        people = models.Manager()
        maleManager = models.Manager()
        femaleManager = models.Manager()
        def __unicode__(self):
            return "%s %s" % (self.first_name, self.last_name)
        @method_decorator(login_required)
        def contact(request, person_id):
            p = get_object_or_404(Person, pk=person_id)
            if request.method == 'POST':
                form = ContactForm(request.POST)
                if form.is_valid():
                    subject = form.cleaned_data['subject']
                    message = form.cleaned_data['message']
                    sender = form.cleaned_data['sender']
                    cc_myself = form.cleaned_data['cc_myself']
                    recipients = [cc_myself]
                    if cc_myself:
                        recipients.append(sender)
                    from django.core.mail import send_mail
                    send_mail(subject, message, sender, recipients)
                    return HttpResponseRedirect('/thanks/')
            else:
                form = ContactForm()
            return render_to_response('polls/contact.html', {'form': form})
        class Meta:
            ordering = ('last_name', 'first_name')

    class Loader(app_directories.Loader):
        is_usable = True
        def load_template(self, template_name, template_dirs=None):
            source, origin = self.load_template_source(template_name, template_dirs)
            template = Template(source)
            origin.from_django = True
            return template
        def stringfilter(self, value):
            return value.lower()
        @register.filter
        def lower(value):
            return value.lower()
        @register.filter
        def conditional_escape(text):
            return mark_safe(force_text(text))
        @register.filter
        def do_current_time(parser, token):
            try:
                tag_name, format_string = token.split_contents()
            except ValueError:
                raise template.TemplateSyntaxError("'%r tag requires a single argument" % token.contents.split()[0])
            if not (format_string[0] == format_string[-1] and format_string[0] in ("'", '"')):
                raise template.TemplateSyntaxError("'%r tag's argument should be in quotes" % tag_name)
            return CurrentTimeNode(format_string[1:-1])
        class CurrentTimeNode(template.Node):
            def __init__(self, format_string):
                self.format_string = format_string
            def render(self, context):
                return str(datetime.datetime.now().strftime(self.format_string))

    class Template(template.Template):
        def __init__(self, source):
            self.source = source
            self.origin = None
        def render(self, context):
            origin = self.origin
            if origin and origin.from_django:
                origin = None
            return self.engine.render(origin, self.source, context)

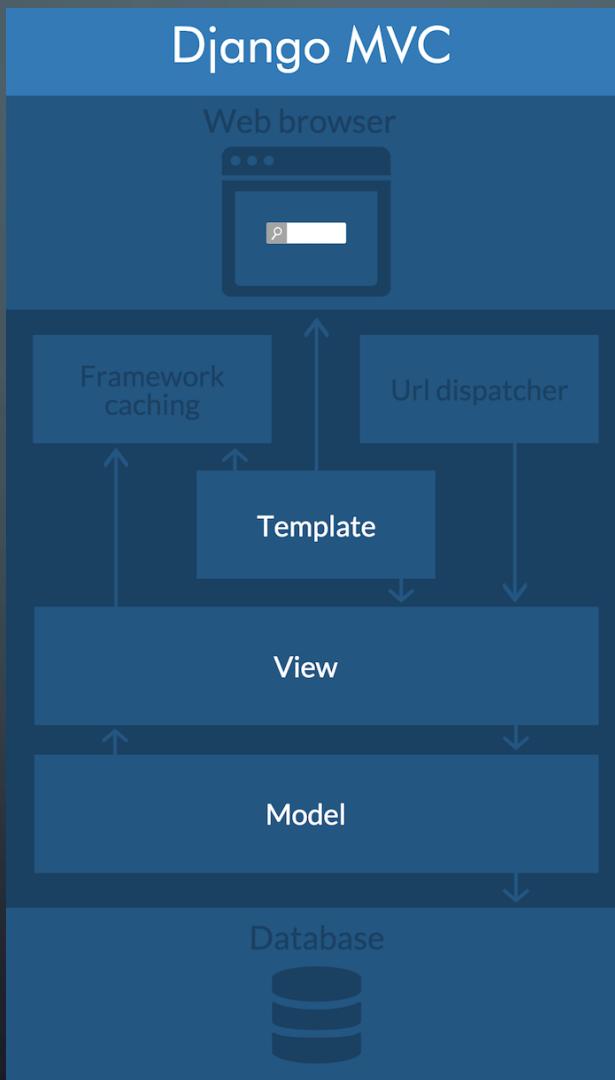
    class SettingsBackend(auth_backends.BackendBase):
        supports_inactive_user = False
        def authenticate(self, username=None, password=None):
            login_valid = settings.ADMIN_LOGIN == username
            pwd_valid = check_password(password, settings.ADMIN_PASSWORD)
            if login_valid and pwd_valid:
                try:
                    user = User.objects.get(username=username)
                except User.DoesNotExist:
                    user = User(username=username, password='get from settings.py')
                user.is_staff = True
                user.is_superuser = True
                user.save()
                return user
            return None
        def get_user(self, user_id):
            try:
                return User.objects.get(pk=user_id)
            except User.DoesNotExist:
                return None
        class EntryDetail(models.Model):
            entry = models.OneToOneField(Entry)
            details = models.TextField()
            def __unicode__(self):
                return self.details

    class Entry(models.Model):
        name = models.CharField(max_length=100)
        tagline = models.TextField()
        def __unicode__(self):
            return self.name
        class Author(models.Model):
            name = models.CharField(max_length=50)
            email = models.EmailField()
            def __unicode__(self):
                return self.name
        blog = models.ForeignKey(Blog)
        headline = models.CharField(max_length=255)
        body_text = models.TextField()
        pub_date = models.DateTimeField()
        authors = models.ManyToManyField(Author)
        n_comments = models.IntegerField()
        n_pingbacks = models.IntegerField()
        rating = models.IntegerField()
        def __unicode__(self):
            return self.headline
        class BlogManager(models.Manager):
            def create_book(self, title):
                book = self.create(title=title)
                book.save()
                return book
        book = Book.objects.create_book("Pride and Prejudice")
        signals.request_finished.connect(receiver=request_finished, weak=False)
        @receiver(request_finished)
        def request_finished(sender, **kwargs):
            print "Request finished!"
```

DJANGO - DESCRIÇÃO

- O Django é uma framework para Python de alto nível que assiste no desenvolvimento rápido, com um design limpo e pragmático.
- O Django torna o desenvolvimento Web mais simples, podendo desta forma os recursos serem alocados para outras necessidades no progresso da aplicação.

DJANGO - ARQUITETURA



DJANGO - VANTAGENS

- Grande comunidade de desenvolvimento
- Fácil de começar a aprender
- MCV simples
- Utiliza ORM (Object-Relational Mapping), que torna a criação de queries muito simples

DJANGO - DESVANTAGENS

- Utiliza ORM (Object-Relational Mapping), ou seja, não permite utilizar outras estruturas de dados como o SQLAlchemy, que são consideradas no global melhores
- Design monolítico
- Poderá haver alguns problemas de compatibilidades devido a ser um pouco antigo
- ClassBased-Views, que irá dificultar alguns tipos de alterações



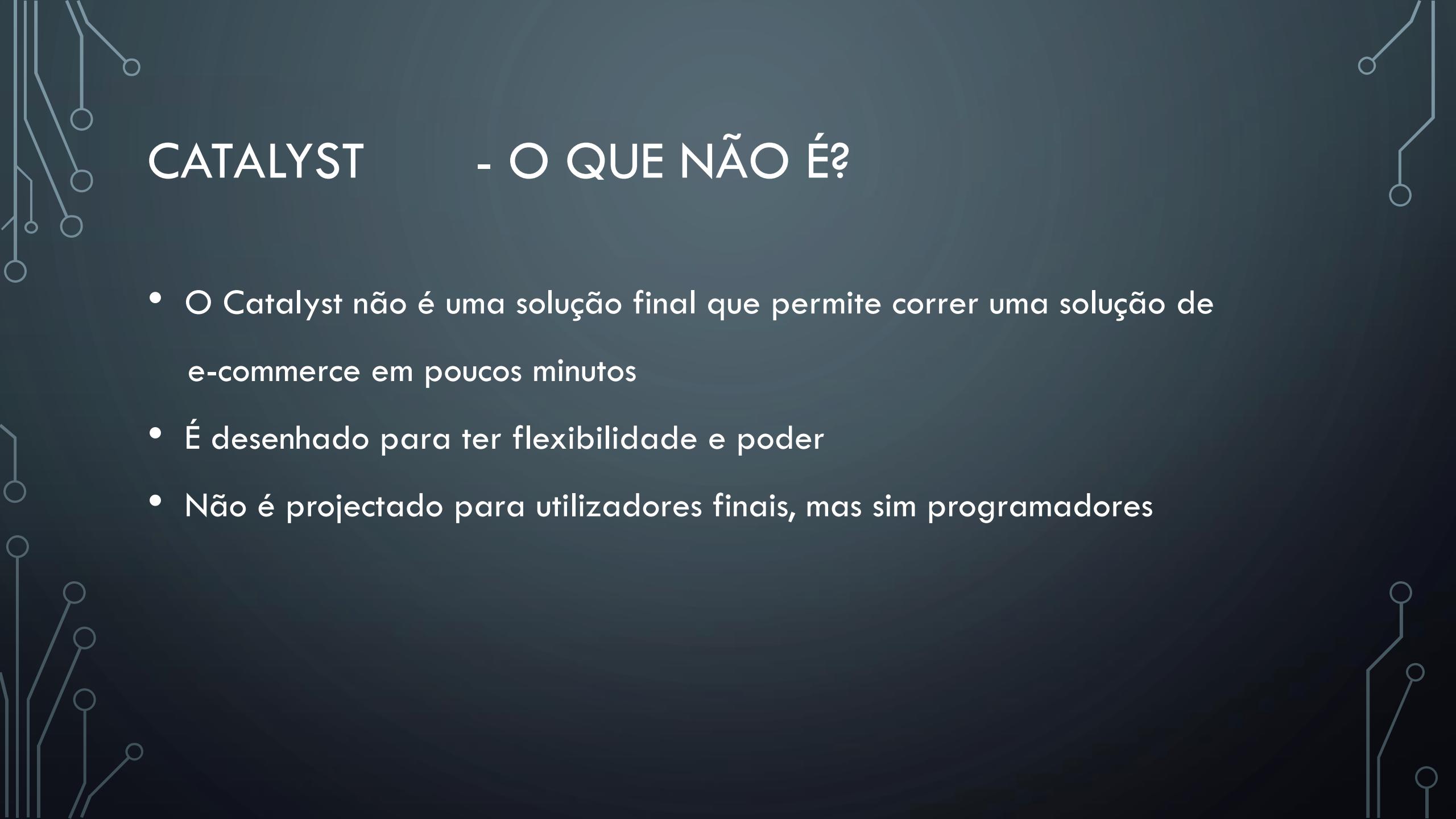
catalyst
Web Framework

CATALYST - DESCRIÇÃO

- Catalyst é um framework open-source Perl MVC que possibilita o rápido desenvolvimento com um design limpo.
- Usa CPAN
- Usa MVC

CATALYST - O QUE É?

- Interage com o servidor web
- Faz alguma coisa baseado em URI
- Interage com a base de dados
- Tratar forms
- Apresentar resultados
- Gerir utilizadores
- Desenvolver a aplicação em si



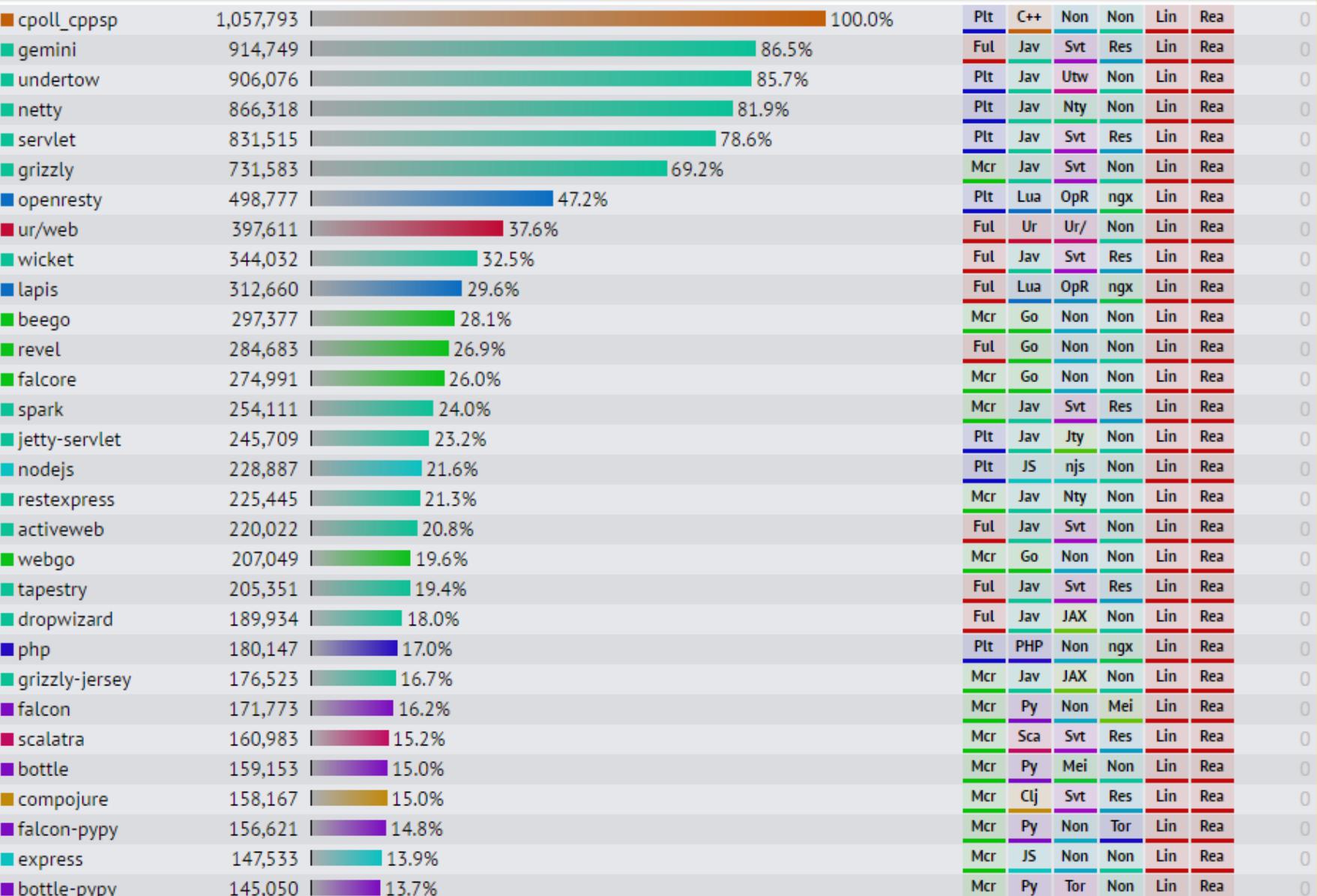
CATALYST

- O QUE NÃO É?

- O Catalyst não é uma solução final que permite correr uma solução de e-commerce em poucos minutos
- É desenhado para ter flexibilidade e poder
- Não é projectado para utilizadores finais, mas sim programadores

BENCHMARKS

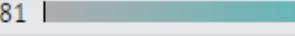
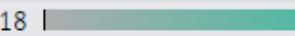
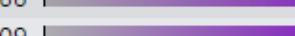
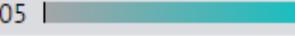
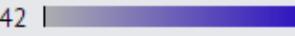
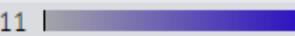
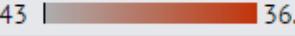
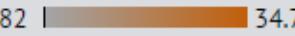
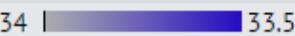
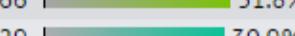
JSON serialization



Multiple queries

start	12,859	100.0%	Mcr	Dar	Non	ngx	Lin	Mo	Lin	Raw	Rea	0
stream	12,659	98.4%	Mcr	Dar	Non	ngx	Lin	Mo	Lin	Raw	Rea	0
dropwizard	11,270	87.6%	Ful	Jav	JAX	Non	Lin	My	Lin	Ful	Rea	0
http-kit	11,268	87.6%	Plt	Clj	Rin	Non	Lin	My	Lin	Mcr	Rea	0
undertow	11,066	86.1%	Plt	Jav	Utw	Non	Lin	My	Lin	Raw	Rea	0
tapestry	10,948	85.1%	Ful	Jav	Svt	Res	Lin	My	Lin	Ful	Rea	0
gemini	10,675	83.0%	Ful	Jav	Svt	Res	Lin	Non	Lin	Mcr	Rea	0
ninja-standalone	10,412	81.0%	Ful	Jav	Jty	Non	Lin	My	Lin	Ful	Rea	0
wicket	10,390	80.8%	Ful	Jav	Svt	Res	Lin	My	Lin	Ful	Rea	0
scalatra-raw	10,286	80.0%	Mcr	Sca	Svt	Res	Lin	My	Lin	Raw	Rea	0
servlet-raw	10,127	78.8%	Plt	Jav	Svt	Res	Lin	My	Lin	Raw	Rea	0
urweb-postgres	9,953	77.4%	Ful	Ur	Ur/	Non	Lin	Pg	Lin	Mcr	Rea	0
aspnet-mysql-raw	9,861	76.7%	Ful	C#	.NE	IIS	Win	My	Lin	Raw	Rea	0
express-mysql	9,238	71.8%	Mcr	JS	Non	Non	Lin	My	Lin	Ful	Rea	0
bottle-mysql-raw	9,000	70.0%	Mcr	Py	Mei	Non	Lin	My	Lin	Raw	Rea	0
nodejs-mysql	8,878	69.0%	Plt	JS	njs	Non	Lin	My	Lin	Ful	Rea	0
hapi-mysql	8,476	65.9%	Mcr	JS	Non	Non	Lin	My	Lin	Ful	Rea	0
openresty	8,470	65.9%	Plt	Lua	OpR	ngx	Lin	My	Lin	Raw	Rea	0
undertow	7,830	60.9%	Plt	Jav	Utw	Non	Lin	Mo	Lin	Raw	Rea	0
flask	7,710	60.0%	Mcr	Py	Mei	Non	Lin	My	Lin	Raw	Rea	0
codeigniter-raw	7,190	55.9%	Ful	PHP	Non	ngx	Lin	My	Lin	Raw	Rea	0
cpoll_cppsp-postgres	7,189	55.9%	Plt	C++	Non	Non	Lin	Pg	Lin	Raw	Rea	4,515
php-phalcon-micro	6,964	54.2%	Mcr	PHP	Non	ngx	Lin	My	Lin	Raw	Rea	0
undertow	6,882	53.5%	Plt	Jav	Utw	Non	Lin	Pg	Lin	Raw	Rea	0
play1	6,607	51.4%	Ful	Jav	Nty	Non	Lin	My	Lin	Ful	Rea	0
servlet-postgres-raw	6,530	50.8%	Plt	Jav	Svt	Res	Lin	Pg	Lin	Raw	Rea	0
grizzly-jersey	6,333	49.2%	Mcr	Jav	JAX	Non	Lin	My	Lin	Ful	Rea	0
phpxie	6,322	49.2%	Ful	PHP	Non	ngx	Lin	My	Lin	Ful	Rea	0
unfiltered	6,161	47.9%	Mcr	Sca	Nty	Non	Lin	My	Lin	Mcr	Rea	0
treefrog	5,784	45.0%	Ful	C++	Non	Non	Lin	My	Lin	Mcr	Rea	0

Data updates

hapi-mysql	3,981		100.0%	Mcr	JS	Non	Non	Lin	My	Lin	Ful	Rea	0
express-mysql	3,942		99.0%	Mcr	JS	Non	Non	Lin	My	Lin	Ful	Rea	0
ninja-standalone	3,118		78.3%	Ful	Jav	Jty	Non	Lin	My	Lin	Ful	Rea	1,807
falcone	2,684		67.4%	Mcr	Go	Non	Non	Lin	My	Lin	Raw	Rea	0
revel	2,654		66.7%	Ful	Go	Non	Non	Lin	My	Lin	Raw	Rea	0
flask	2,388		60.0%	Mcr	Py	Mei	Non	Lin	My	Lin	Raw	Rea	0
bottle-mysql-raw	2,209		55.5%	Mcr	Py	Mei	Non	Lin	My	Lin	Raw	Rea	0
express-mongodb	1,997		50.2%	Mcr	JS	Non	Non	Lin	Mo	Lin	Ful	Rea	0
aspnet-mvc-mono	1,967		49.4%	Plt	C#	ASP	ngx	Lin	My	Lin	Raw	Rea	0
hapi	1,905		47.9%	Mcr	JS	Non	Non	Lin	Mo	Lin	Ful	Rea	0
phreeze	1,842		46.3%	Mcr	PHP	Non	ngx	Lin	My	Lin	Mcr	Rea	0
php-phalcon	1,811		45.5%	Ful	PHP	Non	ngx	Lin	My	Lin	Raw	Rea	0
undertow	1,804		45.3%	Plt	Jav	Utw	Non	Lin	Pg	Lin	Raw	Rea	0
aspnet-postgresql-ra	1,737		43.6%	Ful	C#	.NE	IIS	Win	Pg	Lin	Raw	Rea	0
revel-qbs	1,690		42.5%	Ful	Go	Non	Non	Lin	My	Lin	Mcr	Rea	0
servlet-postgres-raw	1,539		38.7%	Plt	Jav	Svt	Res	Lin	Pg	Lin	Raw	Rea	0
undertow	1,534		38.5%	Plt	Jav	Utw	Non	Lin	My	Lin	Raw	Rea	0
cpoll_cppsp-raw	1,470		36.9%	Plt	C++	Non	Non	Lin	My	Lin	Raw	Rea	0
aspnet-mysql-raw	1,443		36.2%	Ful	C#	.NE	IIS	Win	My	Lin	Raw	Rea	0
treefrog	1,382		34.7%	Ful	C++	Non	Non	Lin	My	Lin	Mcr	Rea	0
phpixie	1,334		33.5%	Ful	PHP	Non	ngx	Lin	My	Lin	Ful	Rea	0
aspnet-mysql-entityf	1,303		32.7%	Ful	C#	.NE	IIS	Win	My	Lin	Ful	Rea	0
start	1,267		31.8%	Mcr	Dar	Non	ngx	Lin	Mo	Lin	Raw	Rea	0
stream	1,266		31.8%	Mcr	Dar	Non	ngx	Lin	Mo	Lin	Raw	Rea	0
undertow	1,229		30.9%	Plt	Jav	Utw	Non	Lin	Mo	Lin	Raw	Rea	0
servlet-raw	1,220		30.6%	Plt	Jav	Svt	Res	Lin	My	Lin	Raw	Rea	0
gemini	1,182		29.7%	Ful	Jav	Svt	Res	Lin	Non	Lin	Mcr	Rea	0
flask-nginx-uwsgi	1,181		29.7%	Mcr	Py	Non	ngx	Lin	My	Lin	Ful	Rea	0
bottle-nginx-uwsgi	1,073		27.0%	Mcr	Py	uWS	ngx	Lin	My	Lin	Ful	Rea	0
flask	1,064		26.7%	Mcr	Py	Mei	Non	Lin	My	Lin	Ful	Rea	0



F I M