

# Taller de JHipster



Adolfo Sanz de Diego  
@asanzdiego



{codemotion}

MAD · NOV 24-25 · 2017

# ¿Quién soy?

**Asesor** técnico docente en la  
Consejería de Educación de la CAM.

Colaboro como **profesor** en la UAH y  
como **formador** en Pronoide.

<https://about.me/asanzdiego>



# HackathonLovers

**Comunidad** de desarrolladores,  
diseñadores, emprendedores...

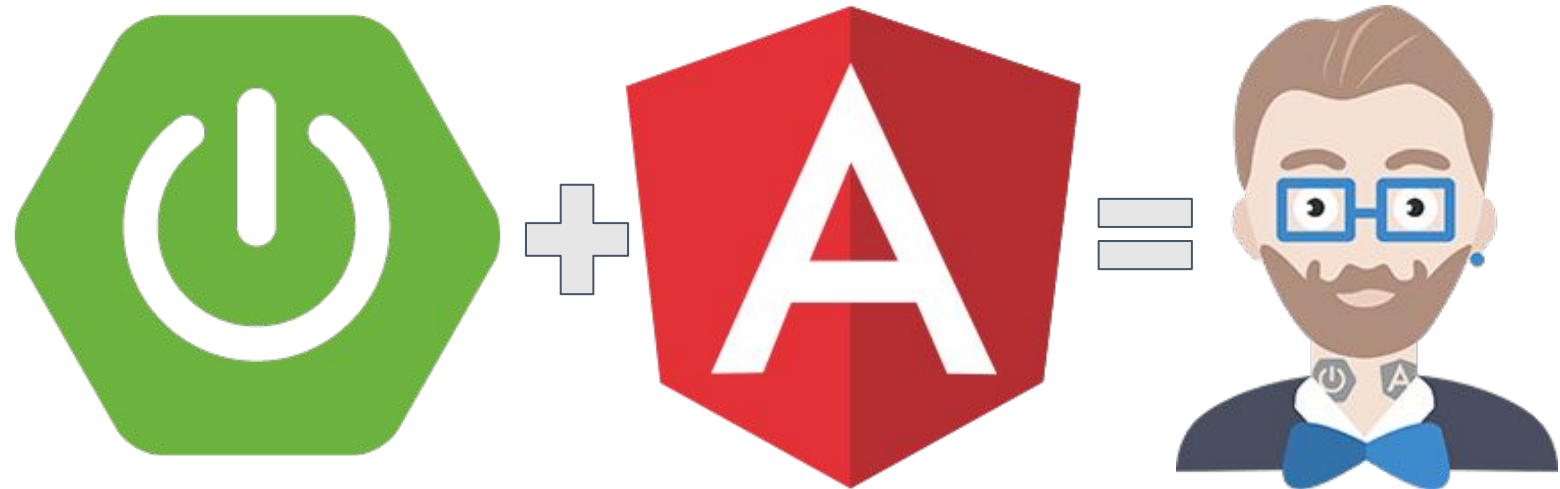
...amantes de los **hackathones**.

<http://hackathonlovers.com>



# ¿Qué es JHipster?

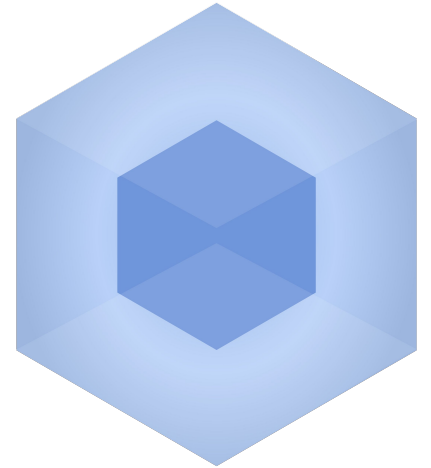
JHipster es una plataforma de desarrollo para generar, desarrollar e implementar aplicaciones web con **SpringBoot y Angular**.





# Tecnologías Front (I)

- Angular
- Bootstrap
- Sass
- Webpack



# Tecnologías Front (II)

- Karma
- Protactor
- Broswersync
- Yarn



# Tecnologías Back (I)

- Spring Boot
- Hibernate
- ElasticSearch
- Gradle



# Tecnologías Back (II)

- Gatlin
- Cucumber
- Liquibase
- Swagger



LIQUIBASE





# Ventajas

- Genera una aplicación muy completa.
- Incluye autenticación, i18n, métricas, logs, etc.
  - El código generado es bastante bueno.
- Es bastante fácil modificar el código generado.
  - Usa un stack bastante probado y conocido.
- Tiene un editor UML muy útil para generar entidades.
  - El equipo que hay detrás es muy activo.
- La comunidad que hay detrás es bastante grande.



# Desventajas

- Conflictos si regeneras una vez modificado el código.
  - Conflictos cuando actualizas la versión.
- Tienes que conocer todo el stack para ser productivo.
  - Puedes pecar de sobre-ingeniería.
- Algunas cosas no están bien optimizadas (ej. imágenes).



# Mi experiencia

- Ideal para prototipado de aplicaciones.
- Ideal para aplicaciones CRUD no muy grandes.
  - Ideal para equipos de desarrollo pequeños.
- Necesario que los desarrolladores sean full-stack.
  - Velocidad de desarrollo alta (al principio).
- Partir de un primer diagrama UML para generar.
  - Luego solo añadir entidades por consola.
- Solo actualizar versión para arreglar fallos críticos.



# Instalación (I)

<http://www.jhipster.tech/installation/>

**Local installation with Yarn (recommended for normal users)**

Quick setup when using Angular



# Instalación (II)

Install Oracle Java 8 (<https://www.java.com/es/download/>)

Install NodeJS LTS (<https://nodejs.org/>)

Install Yarn (<https://yarnpkg.com/en/docs/install>)

```
$ yarn global add yo
```

```
$ yarn global add generator-jhipster
```

```
export PATH="$PATH:`yarn global bin`
```

```
export PATH="$PATH~/.config/yarn/global/node_modules/.bin
```





# Nueva Aplicación

<http://www.jhipster.tech/creating-an-app/>

```
$ mkdir myapplication
```

```
$ cd myapplication/
```

```
$ jhipster
```

Responder a las preguntas...

```
$ ./mvnw
```

 (o 

```
$ ./gradlew
```

) > abrir en el navegador <http://localhost:8080>

```
$ yarn start
```

 (en otra consola) > abrir en el navegador <http://localhost:9000>



# JDL y JDL-Studio


<http://www.jhipster.tech/jdl/>

<http://www.jhipster.tech/jdl-studio/>

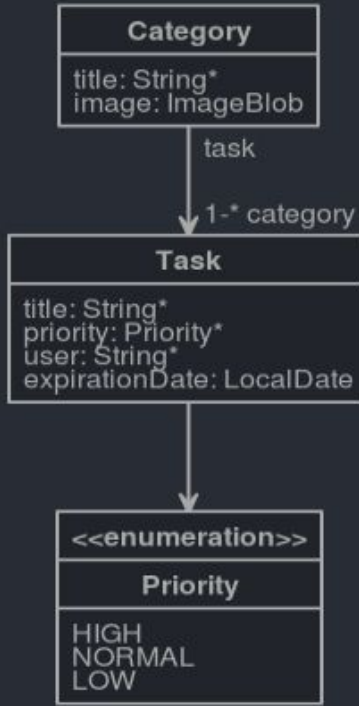
```
$ jhipster import-jdl your-jdl-file.jdl
```



# JDL-Studio

 JDL-Studio

```
enum Priority {  
    HIGH, NORMAL, LOW  
}  
  
entity Task {  
    title String required,  
    priority Priority required,  
    user String required,  
    expirationDate LocalDate  
}  
  
entity Category {  
    title String required,  
    image ImageBlob  
}  
  
relationship OneToMany {  
    Category{task} to Task{category}  
}  
  
paginate Task, Category with pagination
```



```
classDiagram  
    class Category {  
        title String*  
        image ImageBlob  
    }  
    class Task {  
        title String*  
        priority Priority*  
        user String*  
        expirationDate LocalDate  
    }  
    class Priority {  
        <<enumeration>>  
        HIGH  
        NORMAL  
        LOW  
    }  
    Category "1" --> "1-*" Task : task  
    Task --> Priority
```

# Menú administración

- Gestión de usuarios
  - Métricas
  - Salud
- Configuración
- Auditorías
  - Logs
  - API
- Base de datos

(en producción algunos están deshabilitados por defecto)



# Securizar un método del back

./src/main/java/jhipster/web/rest/CategoryResource.java

```
@PostMapping("/categories")
@Timed
@Secured(AuthoritiesConstants.ADMIN)
public ResponseEntity<Category> createCategory(@Valid @RequestBody Category category)
    throws URISyntaxException {
    log.debug("REST request to save Category : {}", category);
    if (category.getId() != null) {
        throw new BadRequestAlertException("Cannot already have an ID", ENTITY_NAME, "idexists");
    }
    Category result = categoryRepository.save(category);
    return ResponseEntity.created(new URI("/api/categories/" + result.getId()))
        .headers(HeaderUtil.createEntityCreationAlert(ENTITY_NAME, result.getId().toString()))
        .body(result);
}
```





# Modificar el menú - solo si logueado

./src/main/webapp/app/layouts/navbar/navbar.component.html

```
<li *ngSwitchCase="true" class="nav-item" routerLinkActive="active" [routerLinkActiveOptions]="{exact: true}">
  <a class="nav-link" routerLink="task" (click)="collapseNavbar()">
    <span>
      <i class="fa fa-home" aria-hidden="true"></i>
      <span jhiTranslate="global.menu.entities.task">Task</span>
    </span>
  </a>
</li>
```

# Modificar el menú - solo si admin

./src/main/webapp/app/layouts/navbar/navbar.component.html

```
<li *jhiHasAnyAuthority="'ROLE_ADMIN'" class="nav-item" routerLinkActive="active"
  [routerLinkActiveOptions]="{exact: true}">
  <a class="nav-link" routerLink="category" (click)="collapseNavbar()">
    <span>
      <i class="fa fa-home" aria-hidden="true"></i>
      <span jhiTranslate="global.menu.entities.category">Category</span>
    </span>
  </a>
</li>
```



# Internalización

./src/main/webapp/i18n/es/priority.json

```
{
  "hipsterApp": {
    "Priority": {
      "null": "",
      "HIGH": "ALTA",
      "NORMAL": "NORMAL",
      "LOW": "BAJA"
    }
  }
}
```

# Filtrar por usuario - Repository

./src/main/java/codemotion/repository/TaskRepository.java

```
@SuppressWarnings("unused")
@Repository
public interface TaskRepository extends JpaRepository<Task, Long> {
    Page<Task> findAllByUser(String user, Pageable pageable);
}
```



# Filtrar por usuario - RestController - createTask

./src/main/java/codemotion/web/rest/TaskResource.java

```
public ResponseEntity<Task> createTask(@Valid @RequestBody Task task) throws URISyntaxException {  
    log.debug("REST request to save Task : {}", task);  
    if (task.getId() != null) {  
        throw new BadRequestAlertException("A new task cannot already have an ID", ENTITY_NAME, "idexists");  
    }  
    if (!SecurityUtils.isCurrentUserInRole(AuthoritiesConstants.ADMIN)) {  
        task.setUser(SecurityUtils.getCurrentUserLogin());  
    }  
    Task result = taskRepository.save(task);  
    return ResponseEntity.created(new URI("/api/tasks/" + result.getId()))  
        .headers(HeaderUtil.createEntityCreationAlert(ENTITY_NAME, result.getId().toString()))  
        .body(result);  
}
```





# Filtrar por usuario - RestController - updateTask

./src/main/java/codemotion/web/rest/TaskResource.java

```
public ResponseEntity<Task> updateTask(@Valid @RequestBody Task task) throws URISyntaxException {  
    if (task.getId() == null) { return createTask(task); }  
    Task taskSaved = taskRepository.findOne(task.getId());  
    if (taskSaved == null || (!SecurityUtils.isCurrentUserInRole(AuthoritiesConstants.ADMIN)  
        && !SecurityUtils.getCurrentUserLogin().equals(taskSaved.getUser())) {  
        return new ResponseEntity<>(HttpStatus.FORBIDDEN);  
    }  
    if (!SecurityUtils.isCurrentUserInRole(AuthoritiesConstants.ADMIN)) {  
        task.setUser(SecurityUtils.getCurrentUserLogin());  
    }  
    Task result = taskRepository.save(task);  
    return ResponseEntity.ok()  
        .headers(HeaderUtil.createEntityUpdateAlert(ENTITY_NAME, task.getId().toString())).body(result);  
}
```



# Filtrar por usuario - Repository - getAllTasks

./src/main/java/codemotion/repository/TaskRepository.java

```
public ResponseEntity<List<Task>> getAllTasks(@ApiParam Pageable pageable) {  
    log.debug("REST request to get a page of Tasks");  
    final Page<Task> page;  
    if (SecurityUtils.isCurrentUserInRole(AuthoritiesConstants.ADMIN)) {  
        page = this.taskRepository.findAll(pageable);  
    } else {  
        page = this.taskRepository.findAllByUser(SecurityUtils.getCurrentUserLogin(), pageable);  
    }  
    HttpHeaders headers = PaginationUtil.generatePaginationHttpHeaders(page, "/api/tasks");  
    return new ResponseEntity<>(page.getContent(), headers, HttpStatus.OK);  
}
```

# Filtrar por usuario - Repository - getTask

./src/main/java/codemotion/repository/TaskRepository.java

```
public ResponseEntity<Task> getTask(@PathVariable Long id) {  
    log.debug("REST request to get Task : {}", id);  
    Task task = taskRepository.findOne(id);  
    if (task != null && !SecurityUtils.isCurrentUserInRole(AuthoritiesConstants.ADMIN)  
        && !SecurityUtils.getCurrentUserLogin().equals(task.getUser())) {  
        return new ResponseEntity<>(HttpStatus.FORBIDDEN);  
    }  
    return ResponseUtil.wrapOrNotFound(Optional.ofNullable(task));  
}
```



# Filtrar por usuario - Repository - deleteTask

./src/main/java/codemotion/repository/TaskRepository.java

```
public ResponseEntity<Void> deleteTask(@PathVariable Long id) {  
    log.debug("REST request to delete Task : {}", id);  
    Task taskSaved = taskRepository.findOne(id);  
    if (taskSaved != null && !SecurityUtils.isCurrentUserInRole(AuthoritiesConstants.ADMIN)  
        && !SecurityUtils.getCurrentUserLogin().equals(taskSaved.getUser())) {  
        return new ResponseEntity<>(HttpStatus.FORBIDDEN);  
    }  
    taskRepository.delete(id);  
    return ResponseEntity.ok()  
        .headers(HeaderUtil.createEntityDeletionAlert(ENTITY_NAME, id.toString())).build();  
}
```



# Filtrar por usuario - Component - ts

./src/main/webapp/app/entities/task/task-dialog.component.ts

```
constructor(  
  ...  
  private principal: Principal  
) {}  
  
ngOnInit() {  
  this.principal.identity().then((account) => {  
    this.task.user = account.login;  
  });  
  ...  
}
```





# Filtrar por usuario - Component - html

./src/main/webapp/app/entities/task/task-dialog.component.html

```
<div *jhiHasAnyAuthority="'ROLE_ADMIN'" class="form-group">
  <label class="form-control-label" jhiTranslate="codemotionApp.task.user" for="field_user">User</label>
  <input type="text" class="form-control" name="user" id="field_user"
    [(ngModel)]="task.user" required />
  <div [hidden]="!(editForm.controls.user?.dirty && editForm.controls.user?.invalid)">
    <small class="form-text text-danger"
      [hidden]="!editForm.controls.user?.errors?.required" jhiTranslate="entity.validation.required">
      This field is required.
    </small>
  </div>
</div>
```

# Cambiar id por nombre categoría - Component - html

./src/main/webapp/app/entities/task/task-dialog.component.html

```
<div class="form-group">
  <label class="form-control-label" jhiTranslate="codemotionApp.task.category"
    for="field_category">Category</label>
  <select class="form-control" id="field_category" name="category" [(ngModel)]="task.category" >
    <option [ngValue]="null"></option>
    <option [ngValue]="categoryOption.id === task.category?.id ? task.category : categoryOption"
      *ngFor="let categoryOption of categories; trackBy: trackCategoryById">{{categoryOption.title}}</option>
  </select>
</div>
```



# Cambiar id por nombre categoría - Component - html

./src/main/webapp/app/entities/task/task.component.html

```
<td>
  <div *ngIf="task.category">
    <a [routerLink]="['../category', task.category?.id ]" >{{task.category?.title}}</a>
  </div>
</td>
```



# Cambiar color según prioridad - Component - ts

./src/main/webapp/app/entities/task/task.component.ts

```
getTrClassFromTask(task: Task) {  
  if (task.priority.toString() === 'HIGH') {  
    return 'table-danger';  
  } else if (task.priority.toString() === 'NORMAL') {  
    return 'table-warning';  
  } else if (task.priority.toString() === 'LOW') {  
    return 'table-success';  
  }  
  return '';  
}
```



# Cambiar color según prioridad - Component - html

./src/main/webapp/app/entities/task/task.component.html

```
<tr *ngFor="let task of tasks ;trackBy: trackId" class="{{getTrClassFromTask(task)}}">
```



# Subida a producción

`./src/main/resources/config/application-prod.yml`

`./mvnw -Pprod clean package`



# Arreglar tests que fallan

`./src/test/java/codemotion/web/rest/TaskResourceIntTest.java`

`// añadir en los métodos que fallan`

`@WithMockUser(username="admin@localhost",authorities={"ROLE_ADMIN"}, password = "admin")`





# Mejorar tests

./src/test/java/codemotion/web/rest/TaskResourceIntTest.java

```
@WithMockUser(username="user@localhost",authorities={"ROLE_USER"}, password = "user")
public void createTaskWithUser() throws Exception {
    ...
    // Validate the Task in the database
    List<Task> taskList = taskRepository.findAll();
    assertThat(taskList).hasSize(databaseSizeBeforeCreate + 1);
    Task testTask = taskList.get(taskList.size() - 1);
    assertThat(testTask.getTitle()).isEqualTo(DEFAULT_TITLE);
    assertThat(testTask.getPriority()).isEqualTo(DEFAULT_PRIORITY);
    assertThat(testTask.getUser()).isEqualTo("user@localhost");
    assertThat(testTask.getExpirationDate()).isEqualTo(DEFAULT_EXPIRATION_DATE);
}
```



# Código del taller

<https://github.com/asanzdiego/codemotion-2017-taller-de-jhipster>



# Feedback



<http://bit.ly/codemotion2017-taller>

# Licencia



Esta obra tiene una licencia:

[Creative Commons-Reconocimiento-CompartirIgual](https://creativecommons.org/licenses/by-sa/4.0/)



**{codemotion}**

MAD · NOV 24-25 · 2017

# ¡Gracias!

