Subir y compartir módulos para la competición

Introducción

¿Tienes una idea para un módulo en Magnolia?

En esta página encontrarás información sobre cómo subir módulos a GitHub, maven y npm. GitHub es uno de los repositorios mas usados para compartir y colaborar en proyectos de código abierto, permite a otros colaborar en tu proyecto reportando bugs o inclusive aportando mejoras. Maven es el gestor de artefactos mas usado en el mundo de Java. npm es el gestor de paquetes de código Javascript mas grande y el usado por defecto en Node.js.

¿Necesitas ideas para módulos?

Aquí tienes algunos tópicos que te pudieran ayudar:

- Integrar valoraciones/comentarios de contenidos.
- Integrar widgets/web components como Google Maps.
- Apps con gráficas, estadísticas, integraciones, etc
- Gestion de carrito / checkout en e-commerce.
- Reserva de sitios libres en sistemas externos: aviones, hoteles, etc.
- Encuestas.

Aquí puedes encontrar un modulo de ejemplo en GitHub:

https://github.com/magnolia-cms/shareable-magnolia

Aquí tienes algunos módulos de ejemplo:

https://github.com/magnolia-cms/language-switcher-magnolia

https://github.com/rah003/neat-tweaks

Nombre

El nombre del módulo debe ser el mismo en Magnolia y en GitHub. El formato debe ser: lo-que-hace-magnolia

Por ejemplo: `language-switcher-magnolia`, `ratings-magnolia`, `shopping-cart-magnolia`.

Licencia

Elige una licencia y agrégala a tu módulo. Recomendamos usar la licencia MIT para los módulos que vayas a compartir públicamente para darle a otros la oportunidad de usar tu módulo en otros proyectos.

README.md

Los ficheros README son imprescindibles para que los desarrolladores encuentren y entiendan tus módulos. Alguien que lea el README de un módulo debería de entenderlo en un minuto gracias a sus pantallazos y descripciones.

Aquí tienes un template para crear tus ficheros README: https://raw.githubusercontent.com/magnolia-cms/shareable-magnolia/master/README.md

Maven

En caso de tener clases de Java, la manera mas sencilla de crear y compilar módulos Java de Magnolia es mediante maven. Magnolia cuenta con repositorios públicos de maven con todas las dependencias necesarias para compilar y desplegar proyectos de Magnolia en versión community y enterprise.

La url del repositorio maven de Magnolia versión comunitaria es: https://nexus.magnolia-cms.com/content/groups/public/

Repositorio local de maven

Para inicializar de forma automática tu repositorio local de maven con el acceso al repositorio de Magnolia puedes ejecutar el siguiente comando:

```
mvn org.sonatype.plugins:nexus-m2settings-maven-plugin:download -DnexusUrl=https://nexus.magnolia-cms.com -
Dusername=anonymous -Dpassword=anonymous -DtemplateId=magnolia-community-public
```

Múltiples repositorios maven

Si ya tenias maven inicializado en tu local, lo mejor es que hagas copia de tu directorio ".m2" y vuelvas a ejecutar el comando de inicialización del repositorio de Magnolia

Dependencias Vaadin

Si tu módulo depende de librerías de Vaadin deberás agregar como repositorio adicional: http://maven.vaadin.com/vaadin-addons/

```
<repositories>
  <repository>
    <id>magnolia.nexus.public</id>
    <url>https://nexus.magnolia-cms.com/content/groups/public/</url>
    <releases>
      <enabled>true</enabled>
    </releases>
    <snapshots>
      <enabled>true</enabled>
    </snapshots>
  </repository>
  <repository>
    <id>vaadin-addons</id>
    <url>http://maven.vaadin.com/vaadin-addons/</url>
    <releases>
      <enabled>true</enabled>
    </releases>
    <snapshots>
      <enabled>true</enabled>
    </snapshots>
  </repository>
</repositories>
```

Arquetipo maven para proyectos Magnolia

Magnolia provee arquetipos para distintos tipos de proyectos Java.

Para obtener el listado de arquetipos disponibles, hay que ejecutar lo siguiente:

```
mvn org.apache.maven.plugins:maven-archetype-plugin:2.4:generate -DarchetypeCatalog=https://nexus.magnolia-cms.
com/content/groups/public/
```

Para crear un proyecto para un módulo de Magnolia seleccionar la opción 3:

```
3: https://nexus.magnolia-cms.com/content/groups/public/ -> info.magnolia.maven.archetypes:magnolia-module-
archetype (An archetype to create basic Magnolia modules)
```

Si en cambio necesitas crear un proyecto completo de Magnolia (para desarrollar/probar tu modulo en una instancia de Magnolia community), seleccionar la opción 2:

```
2: https://nexus.magnolia-cms.com/content/groups/public/ -> info.magnolia.maven.archetypes:magnolia-project-archetype (An archetype to create a Magnolia project (a parent pom and a webapp))
```

Aquí puedes encontrar un ejemplo del pom de un modulo de ejemplo: https://github.com/ebguilbert/tours-field/blob/master/pom.xml

Git

Inicializa un repositorio git en el directorio de tu módulo. El código puede estar alojado en cualquier servidor git. Aquí estamos usando GitHub como ejemplo.

Si creas una cuenta de https://github.com y creas el repositorio mediante la web, la página te proveerá con instrucciones muy útiles para inicializar y conectar tu proyecto con el repositorio git.

Por favor rellena la descripción del repositorio indicando que es un módulo de Magnolia, y no olvides agregar los tópicos:

- `rd-magnolia-community-2019`
- `magnolia`
- `magnolia-component`

NPM

En el caso de tener un proyecto de front-end basado en Javascript y siguiendo la estructura de un Light Module, lo recomendado para distribuir el código es registrarlo en npm.

> **Magnolia CLI**
>
> Si no conoces la estructura de un Light Module, puedes utilizar Magnolia CLI para crear la estructura del módulo automaticamente. Mas info: https://documentation.magnolia-cms.com/display/DOCS60/Magnolia+CLI+walkthrough

npm init

El primer paso para registrar un proyecto en npm es inicializar el proyecto. Para esto se debe invocar:

```
npm init
```

Aceptar todas las opciones por defecto. Esto creará un package.json que represntará el descriptor del paquete en npm.

Una vez se haya creado el fichero package.json, se deben agregar las siguientes keywords para asistir la automatización de las búsquedas en npm:

- `magnolia-light-module` (Requerido)
- `magnolia-component` (Opcional - para un componente simple)
- `magnolia-kit` (Opcional - para un conjunto de componentes)

Ejemplo:

```
"keywords": [
    "magnolia-light-module",
    "magnolia-component"
  ],
```

module.yaml

Es necesario agregar un fichero module.yaml, que representará el descriptor del módulo en Magnolia. En este fichero se define la versión del módulo y opcionalmente las dependencias a otros modulos.

Ejemplo: https://raw.githubusercontent.com/magnolia-cms/shareable-magnolia/master/module.yaml

Publicar en npm

Para publicar un módulo de Magnolia en npm, se debe invocar:

```
npm publish
```

Este comando interactivo permitirá crear un usuario en npm en caso de no tenerlo aún.

Se puede consultar el paquete publicado en `https://www.npmjs.com/package/<light module name>`

El modúlo estará disponible para búsqueda en la web de npm 😎

> **Mas info**
>
> Mas info sobre light modules en Magnolia y npm: How To Open Source a Light Module

Compartirlo

Hazte conocer y comparte tu código de Magnolia con el mundo. Menciona @magnoliacms_es en Twitter e incluye el hashtag #RDMagnolia y nosotros lo compartiremos con nuestros seguidores. También puedes crear tu propio blog acerca del módulo, o inclusive puedes contactarnos para publicarlo en los blogs de Magnolia.