# Introducción

Este documento presenta la previsión de las líneas de código del proyecto Jugador número 12. Se ha hecho analizando en la medida de lo posible proyectos existentes de envergadura similar, pero sobre todo tomando como punto de partida las líneas de código existentes a comienzos del mes de diciembre del 2012.

En ese momento, la funcionalidad era mínima y se ha extrapolado una aproximación para implementar la funcionalidad buscada. Se intenta una aproximación para cada lenguaje de programación que se va a usar en el proyecto

# Índice

[1. Análisis de la previsión de las líneas de código.](#h.a8e2sbpde1ha)

[1.1 CSS](#h.6h9od7ay62t8)

[1.1.1. Menús](#h.6wzptw8jqq6c)

[1.1.2. Divisiones](#h.4roo8zz0wmv4)

[1.1.3. Estilos](#h.1jia23tyav6i)

[1.1.4. CSS del framework](#h.cckgu0xxwd4f)

[1.1.5. Clase principal](#h.tq4ebfok8py6)

[1.2. Javascript](#h.lfkazuca9efd)

[1.2.1 Validación de formularios](#h.9xlp4u1m12hg)

[1.2.2. Uso de Ajax](#h.x8h5grnwyuve)

[1.2.3. Llamadas a funciones del framework](#h.bkwwrkdo67aq)

[1.3. MySQL](#h.wr1eyfj521hp)

[1.3.1. Estructura básica de las tablas](#h.ginleuohhfa5)

[1.3.2. Relleno inicial de datos](#h.bg2lgyra8ad2)

[1.4. HTML](#h.t4g6ua7wufkr)

[1.4.1 Layouts](#h.dwbl4mbny82e)

[1.4.2. Vistas](#h.rjk4t1mm80v)

[1.5. PHP](#h.qgzxzpbebno7)

[1.5.1. Controladores](#h.hx0xwdn9wqjk)

[1.5.2. Modelos](#h.nprjzpy6yio8)

[1.5.3. Componentes adicionales](#h.npbxjzifsst7)

[1.5.4. Añadido extra](#h.dggf3geus4s7)

[2. Conclusiones](#h.ccwpxdvys3m8)

[3. Anexos](#h.d7gtavz9uty8)

[Previsión de las lineas de código del proyecto](#h.agsq9gjn03ag)

# 1. Análisis de la previsión de las líneas de código.

En primer lugar debemos resaltar que, dado el uso de un framework de PHP (Yii), la cantidad de líneas de código puede verse considerablemente reducida en comparación con otras aplicaciones reales. Hay que mencionar también que trataremos de redondear a lo alto las líneas de código calculadas ya que se supondrá que inicialmente no habrá optimizaciones en el código y puede haber fragmentos del sistema poco refinados o con redundancias.

## 1.1 CSS

Analizaremos en primer lugar una de las partes más sencillas, que se corresponde a los

documentos de hojas de estilo (CSS). Dado que son comunes a toda la aplicación (en concreto

a las vistas) trataremos de hacer una estimación basándonos en divisiones de funcionalidades:

### 1.1.1. Menús

Partiendo de que la aplicación contiene actualmente un sencillo menú con una funcionalidad más que suficiente para la misma, nos basaremos en este diseño para realizar una estimación. Actualmente esta sección consta de 134 líneas de código CSS. Teniendo en cuenta que tiene actualmente incluida funcionalidad para submenús (de los cuales no haremos uso), podemos establecer una cota inferior en unas 100 LDC. Suponiendo que podamos añadir aún más funcionalidad/decoración que la actual, pasaríamos a una cota superior de 180 LDC.   
Esto nos ofrece una media de LDC para los menús de **140**.

### 1.1.2. Divisiones

Esta sección tratará de las CSS asociadas a la distribución de elementos. Dado que en el mockup se cuenta actualmente con unas 80 LDC para la distribución más básica, daremos en este apartado un margen mucho más amplio. Podemos suponer que se rondarán las 200 LDC para maquetación básica y le añadiremos un extra de 50 líneas para refinado.  
Esto suma un total esperado de **250** líneas de código CSS.

### 1.1.3. Estilos

Apartado que corresponde a la definición de estilos (color, fuente, etc.) para las distribuciones y elementos anteriores. Actualmente consta de tan sólo 20 líneas de código, con lo que ampliaremos el margen hasta **150** líneas ya que pueden ser más que suficientes para nuestros objetivos.

### 1.1.4. CSS del framework

Actualmente, y posiblemente sea lo único necesario, se hace uso de los elementos del widget de formularios de Yii.  
Estos ocupan un total aproximado de **130** LDC.

### 1.1.5. Clase principal

Clase principal: se encargará de reunir todos los elementos CSS a través de sentencias “Import”. Su número de líneas de código estará situado cerca de 10.

## 1.2. Javascript

Usado sobre todo para validaciones o funciones sencillas a ejecutar en el lado del cliente. Incluiremos en esta sección las LDC de Ajax. Puesto que haremos uso del framework de Yii, la mayor parte de este apartado vendrá dado por funciones internas ya creadas, por lo que el número de LDC esperado será mucho menor al de una aplicacion real creada sin ayuda de librerías.

### 1.2.1 Validación de formularios

Conformará una gran parte de este tipo de código. Supondremos una cantidad media de 5 LDC por campo validado y una media de 20 campos o secciones a validar. El número de validaciones es reducido ya que se ha tratado de eliminar las realizadas por el framework Yii.  
Previsión de **100** LDC.

### 1.2.2. Uso de Ajax

Uso de Ajax: en situaciones como la realización de acciones durante un partido. Dado que no viene implementado por defecto en Yii, este apartado aumentará bastante la cantidad de LDC de Javascript. Supondremos una media de 10 subprogramas para llevar a cabo toda la funcionalidad necesaria asíncronamente y unas 50 líneas de código por cada uno.  
Esto suma un total de **500** LDC.

### 1.2.3. Llamadas a funciones del framework

Llamadas a funciones del framework: será una mínima parte del total, pero añadiremos un extra de **40** LDC para esta sección.

## 1.3. MySQL

Crearemos una estimación basándonos en dos parámetros principales:

### 1.3.1. Estructura básica de las tablas

Partiendo de que el prototipo se ha creado a partir de un archivo de unas 120 LDC. Suponiendo que dicha estructura se corresponda a un 60% de la total, la composición final supondría un archivo de unas **200** líneas de código SQL.

### 1.3.2. Relleno inicial de datos

Añadiremos un extra de **500** LDC ya que será necesario inicializar determinados elementos como los equipos o habilidades.

## 1.4. HTML

Conformará el grueso de la parte de presentación, por lo que dividiremos esta sección

en los siguientes apartados:

### 1.4.1 Layouts

Formados en su mayor parte por código HTML. Su estimación se basará en los existentes actualmente en el mockup.   
Éstos contienen alrededor de **120** LDC.

### 1.4.2. Vistas

Supondremos una media de 70 líneas de código para cada una, y el uso de un total aproximado de 50 vistas.

Esto nos da un total de **3500** LDC.

## 1.5. PHP

Esta sección conformará el grueso de la aplicación. Al igual que las anteriores la

dividiremos en diferentes apartados:

### 1.5.1. Controladores

De nuevo, realizando suposiciones, partiremos de los siguientes parámetros: 8 controladores de media, 5 acciones por cada uno y 2 funciones auxiliares creadas. Una posible estimación en líneas básicas de código por controlador es de 70, 50 por acción y 20 por función auxiliar creada. Teniendo todo esto en cuenta.

Obtenemos un total aproximado de **3000** LDC.

### 1.5.2. Modelos

Actualmente contamos con 13 modelos para la estructura básica. Teniendo en cuenta que aún faltan modelos para el sistema de mensajes privados y posiblemente la organización de roles de usuario, tomaremos una media de 16 modelos con 150 LDC para cada uno.

Esto nos da una medida aproximada de **2400** LDC.

### 1.5.3. Componentes adicionales

Aquí incluiremos elementos adicionales como clases auxiliares, componentes y scripts.

1. Clases auxiliares: el grueso de las mismas estará compuesto por los Singleton usados para la funcionalidad de las acciones del juego. Suponiendo una media de 50 clases (con una acción asociada) y 70 líneas de código para cada una obtenemos un total de **3500** líneas de código. Se ha engrosado un poco este resultado para tener en cuenta posibles clases más pequeñas como Validadores personalizados.
2. Componentes: conformarán una muy pequeña parte pero se mencionarán para mayor precisión. Actualmente se hace uso tan sólo del archivo UserIdentity.php, que consta de 30 LDC. Suponiendo la creación de algún componente extra, aumentaremos esta cifra hasta las **100** LDC.
3. Scripts: de nuevo, conformarán gran parte de la funcionalidad del sistema. Vamos a suponer el uso de un total de 3 scripts pequeños (unas 200 LDC) y 1 más grande para simular partidos (unas 400 LDC). Esto suma un total de **1000** líneas de código para este apartado.

### 1.5.4. Añadido extra

Sumamos una pequeña cantidad de 1300 líneas de código para cubrir posibles ampliaciones o funciones auxiliares inesperadas así como algunos huecos que pudiesen haber quedado en secciones anteriores.

# 2. Conclusiones

Tras todo este estudio, los cálculos finales para la estimación de líneas de código de la aplicación se encuentran en la hoja de cálculo del [anexo1](#id.t9daqlk7nxvo). Los redondearemos ligeramente con respecto a los obtenidos con anterioridad.

# 3. Anexos

1. Manuel Artero, [Hoja de cálculo con el reparto de las lineas de código estimadas](https://docs.google.com/spreadsheet/ccc?key=0Ao09kH8yz2wzdDUxcGdDRlVoWGtrUnp2NUFkOEhSWHc#gid=0) : <https://docs.google.com/spreadsheet/ccc?key=0Ao09kH8yz2wzdDUxcGdDRlVoWGtrUnp2NUFkOEhSWHc#gid=0>