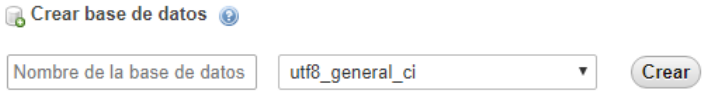
Poniendo MySQL en UTF-8

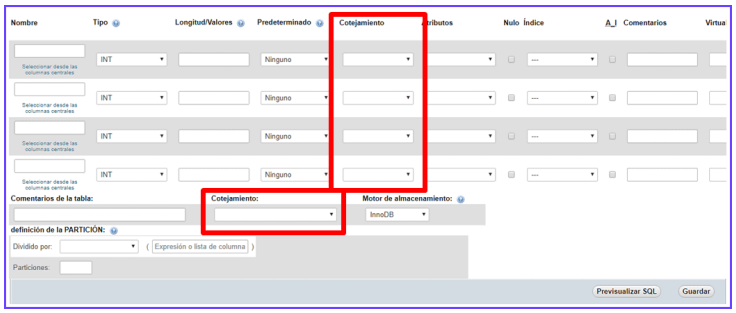
La codificación de contenidos cuando se desarrolla una aplicación que requiere el uso de bases de datos puede llegar a convertirse en una de las peores pesadillas de un desarrollador, sobre todo si es algo en lo que no has pensado desde las primeras fases iniciales previas al inicio del desarrollo. Motores de bases de datos tan populares y difundidos como MySQL pueden dar muchísimos problemas a la hora de grabar datos, de recuperarlos y de renderizarlos en vistas, si no tenemos en cuenta la codificación.
La codificación de datos más adecuada en la mayor parte de los casos es UTF-8, ya que implementa todos los caracteres que se puedan usar en cualquier idioma, así como gran cantidad de guarismos que, sin pertenecer a un idioma específico, son de uso relativamente común, como signos matemáticos u otros. Para implementar otros caracteres extremadamente especiales hay otras codificaciones adecuadas, pero eso es tan poco habitual que ni lo vamos a comentar.
El problema es que, cuando se crea una base de datos, tabla o campo, el motor de MySQL (dependiendo de la versión, y del entorno de desarrollo) tiende a emplear alguna codificación de tipo latin-n-xxxxxxxx o similar. De momento, parece irrelevante, pero nada más lejos de la verdad. A menos que quieras, cada vez que grabas o recuperas datos, tener que perder horas de desarrollo y toneladas de paciencia empleando las temidas y obsoletas entidades HTML, es mejor que te asegures de que tu motor y tus datos empleen UTF-8.

PREPARANDO EL MOTOR]
Esta es la parte más fundamental a la hora de hacer que puedas gestionar tus bases de datos en UTF-8, y la que más desarrolladores pasan por alto. Debemos configurar el motor de MySQL para que use esta codificación por defecto en todas sus operaciones. Para ello, tenemos que bucear en la instalación de MySQL. Esta puede variar mucho, según como tengas montado tu entorno de desarrollo, la plataforma en la que estés trabajando, etc. Lo que importa es que tenemos que buscar un archivo que se llama my.ini o, en determinadas instalaciones, my.cnf. En mi caso concreto, por ejemplo, que trabajo con Xampp en una plataforma Windows, la ruta es C:/xampp/mysql/bin/my.ini. En tu caso, si usas Wamp, o Mamp en un Mac, o Linux, el archivo buscado puede estar en otra parte.
Abriremos el fichero indicado con un editor de texto plano. Puedes usar el NotePad ++, el SublimeText, el Visual Studio Code, o el que tú desees. Incluso, si te va el masoquismo, puedes usar el bloc de notas de Windows. El caso es que tienes que buscar la clave [mysqld. Bajo la misma ya hay algunos datos de configuración. Tienes que buscar que existan, en este apartado, las siguientes líneas:
character-set-filesystem=UTF8
character-set-server=UTF8
default-collation=UTF8_general_ci
default-character-set=UTF8
Si falta alguna de ellas (al menos dos seguro que te faltan), escríbelas ahora.
A continuación tienes que buscar la clave [client. Bajo ella debes escribir la línea siguiente:
default-character-set=UTF8
Por último, busca la clave [mysqldump. Bajo ella escribe la siguiente línea (la misma que en el apartado anterior):
default-character-set=UTF8
A continuación, graba el archivo con los cambios y reinicia el servicio MySQL, para que adopte la nueva configuración.
LAS ESTRUCTURAS DE DATOS]
Ya tenemos el motor de MySQL listo para trabajar en UTF-8. Ahora llega el momento de tener este tema en cuenta cuando crees una nueva base de datos, tabla o campo, o para modificar los ya existentes. Esto último puede llegar a ser muy engorroso, casi desesperante y, dependiendo de las estructuras que tengas, consumir gran cantidad de horas. Por suerte, sólo hay que hacerlo una vez.
Supondremos que gestionas tus bases de datos con PHPMyAdmin. Es una herramienta muy habitual y cómoda, que muchos

empleamos a diario. Cuando vamos a crear una nueva base de datos, al lado del campo de texto para escribir el nombre de la base de datos aparece un desplegable para seleccionar la codificación. Si has configurado tu motor como te he comentado en el apartado anterior, por defecto debería tener seleccionado el valor utf8_general_ci. Si no es así, selecciónalo ahora. Te deberá quedar como se ve a continuación:



Por cierto. La terminación _ci significa case insensitive, es decir, que los datos que se graben serán localizados, tanto si están en mayúsculas como en minúsculas. Este es el funcionamiento de las bases de datos MySQL.
Una vez creada la base de datos llega el momento de crear una tabla. Al hacerlo, y tras darle un nombre y el número de campos llegamos a la siguiente pantalla:



Observa las dos zonas rodeadas en rojo. Tenemos que establecer el cotejamiento general de la tabla y, en cada campo que vaya a tener datos de texto (CHAR, VARCHAR, TEXT, etc) también deberemos establecerlo. Lo pondremos en utf8_general_ci.
En el caso de las estructuras ya existentes, las modificaremos con la nueva codificación. Aún así, algunos datos antiguos conservarán la codificación que ya tuvieran grabada. Eso es un problema más difícil de solucionar. Ya tenemos que resolverlo programáticamente. Deberemos leer los contenidos, cambiarles la codificación (en PHP podemos usar mb_convert_encoding(), por ejemplo) y actualizar cada registro. Si tu tabla tiene gran cantidad de datos, o estructuras muy complejas, la tarea puede ser bastante ardua. Por eso, es mejor emplear UTF-8 para todo desde el principio.