EBook Gratis

APRENDIZAJE encoding

Free unaffiliated eBook created from
Stack Overflow contributors.

#encoding

Tabla de contenido

Acerca de

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: encoding

It is an unofficial and free encoding ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official encoding.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Capítulo 1: Empezando con la codificación

Observaciones

¿Qué es una codificación y cómo funciona?

Una computadora no puede almacenar letras o cualquier otra cosa, almacena bits. El bit puede ser 0 o 1 ("sí" / "no", "verdadero" / "falso"; por lo tanto, estos formatos se denominan binarios). Para usar estos bits se requieren algunas reglas, para convertir los bits en algún contenido. Estas reglas se denominan *codificaciones* , donde las secuencias de 1/0 bits representan ciertos caracteres. Una secuencia de 8 bits se llama *byte* .

Las codificaciones funcionan como tablas, donde cada carácter está relacionado con un byte específico. Para codificar algo en codificación ASCII, se deben seguir las entradas de derecha a izquierda, buscando bits relacionados con los caracteres. Para decodificar una cadena de bits en caracteres, uno sustituye los bits por letras de izquierda a derecha.

Los bytes se pueden representar en diferentes formatos: por ejemplo, `10011111` en binario es `237` en octal, `159` en decimal y `9F` en formatos hexadecimales.

¿Cuál es la diferencia entre las diferentes codificaciones?

La primera codificación de caracteres como ASCII de la era anterior a 8 bits usó solo 7 bits de 8. ASCII se usó para codificar el idioma inglés con todas las 26 letras en mayúsculas y minúsculas, números y muchos signos de puntuación. ASCII no pudo cubrir otros idiomas europeos con todas las letras `ö-ß-é-å` , por lo que se desarrollaron codificaciones que utilizaban el bit 8º de un byte para cubrir otros 128 caracteres.

Pero un byte no es suficiente para representar idiomas con más de 256 caracteres, por ejemplo, chino. El uso de dos bytes (16 bits) permite la codificación de 65,536 valores distintos. Tales codificaciones como BIG-5 separan una cadena de bits en bloques de 16 bits (2 bytes) para codificar caracteres. Las codificaciones de múltiples bytes tienen la ventaja de ser eficientes en el espacio, pero la desventaja de que las operaciones como la búsqueda de subcadenas, comparaciones, etc., tienen que decodificar los caracteres a puntos de código Unicode antes de que se puedan realizar dichas operaciones (aunque hay algunos accesos directos).).

Otro tipo de codificación es el que tiene un número variable de bytes por carácter, como los estándares UTF. Estos estándares tienen algún tamaño de unidad, que para UTF-8 es de 8 bits, para UTF-16 es de 16 bits, y para UTF-32 es de 32 bits. Y luego el estándar define algunos de los bits como indicadores: si están configurados, entonces la siguiente unidad en una secuencia de unidades debe considerarse parte del mismo carácter. Si no están configurados, esta unidad representa solo un carácter por completo (por ejemplo, el inglés ocupa solo un byte, y por eso la codificación ASCII se asigna por completo a UTF-8).

¿Qué es el Unicode?

Unicode si es un conjunto de caracteres enorme (que se dice de una manera más comprensible,

una tabla) con 1,114,112 puntos de código, cada uno de ellos representa una letra, un símbolo u otro carácter específico. Usando Unicode, puede escribir un documento que contenga teóricamente cualquier lenguaje utilizado por las personas.

Unicode no es una codificación, es un conjunto de puntos de código. Y hay varias formas de codificar puntos de código Unicode en bits, como UTF-8, -16 y -32.

Examples

Instalación o configuración

Instrucciones detalladas sobre cómo configurar o instalar la codificación.

¿Cómo detectar la codificación de un archivo de texto con Python?

Hay un paquete útil en Python - chardet, que ayuda a detectar la codificación utilizada en su archivo. En realidad, no hay ningún programa que pueda decir con 100% de confianza qué codificación se utilizó, por eso chardet ofrece la codificación con la mayor probabilidad de codificación con el archivo. Chardet puede detectar las siguientes codificaciones:

- ASCII, UTF-8, UTF-16 (2 variantes), UTF-32 (4 variantes)
- Big5, GB2312, EUC-TW, HZ-GB-2312, ISO-2022-CN (chino tradicional y simplificado)
- EUC-JP, SHIFT_JIS, CP932, ISO-2022-JP (japonés)
- EUC-KR, ISO-2022-KR (coreano)
- KOI8-R, MacCyrillic, IBM855, IBM866, ISO-8859-5, windows-1251 (cirílico)
- ISO-8859-2, windows-1250 (húngaro)
- ISO-8859-5, windows-1251 (búlgaro)
- windows-1252 (inglés)
- ISO-8859-7, windows-1253 (griego)
- ISO-8859-8, windows-1255 (hebreo visual y lógico)
- TIS-620 (tailandés)

Puedes instalar chardet con un comando pip :

```
pip install chardet
```

Después puede usar chardet en la línea de comando:

```
% chardetect somefile someotherfile
somefile: windows-1252 with confidence 0.5
someotherfile: ascii with confidence 1.0
```

o en python:

```
import chardet
rawdata = open(file, "r").read()
result = chardet.detect(rawdata)
charenc = result['encoding']
```

Lea Empezando con la codificación en línea:
https://riptutorial.com/es/encoding/topic/6755/empezando-con-la-codificacion

Creditos

S. No	Capítulos	Contributors
1	Empezando con la codificación	Community, vlad.rad