

Taller Ext2

Gonzalo Pablo Fernández

Departamento de Computación, FCEyN,
Universidad de Buenos Aires, Buenos Aires, Argentina

Sistemas Operativos, segundo cuatrimestre de 2017

(2) Taller de Sistemas de Archivos

- Hoy vamos a programar ext2

(2) Taller de Sistemas de Archivos

- Hoy vamos a programar ext2
- Puntualmente, leer el contenido de un archivo de texto en un disco virtual

(2) Taller de Sistemas de Archivos

- Hoy vamos a programar ext2
- Puntualmente, leer el contenido de un archivo de texto en un disco virtual
- ¿Qué tenemos para hacerlo?

(2) Taller de Sistemas de Archivos

- Hoy vamos a programar ext2
- Puntualmente, leer el contenido de un archivo de texto en un disco virtual
- ¿Qué tenemos para hacerlo?
 - Lo que aprendimos en la teoría sobre ext2

(2) Taller de Sistemas de Archivos

- Hoy vamos a programar ext2
- Puntualmente, leer el contenido de un archivo de texto en un disco virtual
- ¿Qué tenemos para hacerlo?
 - Lo que aprendimos en la teoría sobre ext2
 - Lo que aprendimos en la práctica sobre ext2

(2) Taller de Sistemas de Archivos

- Hoy vamos a programar ext2
- Puntualmente, leer el contenido de un archivo de texto en un disco virtual
- ¿Qué tenemos para hacerlo?
 - Lo que aprendimos en la teoría sobre ext2
 - Lo que aprendimos en la práctica sobre ext2
 - Un disco al cual podemos acceder a cualquiera de sus bloques

(3) El disco

- ¿Qué es? Un montón de bits agrupados en bloques

(3) El disco

- ¿Qué es? Un montón de bits agrupados en bloques
- A cada bloque lo accedo con su LBA (Logical Block Addressing)

(3) El disco

- ¿Qué es? Un montón de bits agrupados en bloques
- A cada bloque lo accedo con su LBA (Logical Block Addressing)
- API de HDD:

```
int read(unsigned int lba, unsigned char * buffer);  
int write(unsigned int lba, unsigned char * buffer);
```

(3) El disco

- ¿Qué es? Un montón de bits agrupados en bloques
- A cada bloque lo accedo con su LBA (Logical Block Addressing)
- API de HDD:

```
int read(unsigned int lba, unsigned char * buffer);  
int write(unsigned int lba, unsigned char * buffer);
```
- Con lo que aprendieron y la API para leer, **a programar!**

(3) El disco

- ¿Qué es? Un montón de bits agrupados en bloques
- A cada bloque lo accedo con su LBA (Logical Block Addressing)
- API de HDD:

```
int read(unsigned int lba, unsigned char * buffer);  
int write(unsigned int lba, unsigned char * buffer);
```
- Con lo que aprendieron y la API para leer, **a programar!**
- ¿Dudas?

(3) El disco

- ¿Qué es? Un montón de bits agrupados en bloques
- A cada bloque lo accedo con su LBA (Logical Block Addressing)
- API de HDD:

```
int read(unsigned int lba, unsigned char * buffer);  
int write(unsigned int lba, unsigned char * buffer);
```
- Con lo que aprendieron y la API para leer, **a programar!**
- ¿Dudas?
- ¿Qué tamaño tiene el disco?

(3) El disco

- ¿Qué es? Un montón de bits agrupados en bloques
- A cada bloque lo accedo con su LBA (Logical Block Addressing)
- API de HDD:

```
int read(unsigned int lba, unsigned char * buffer);  
int write(unsigned int lba, unsigned char * buffer);
```
- Con lo que aprendieron y la API para leer, **a programar!**
- ¿Dudas?
- ¿Qué tamaño tiene el disco?
Ni idea

(3) El disco

- ¿Qué es? Un montón de bits agrupados en bloques
- A cada bloque lo accedo con su LBA (Logical Block Addressing)
- API de HDD:

```
int read(unsigned int lba, unsigned char * buffer);  
int write(unsigned int lba, unsigned char * buffer);
```
- Con lo que aprendieron y la API para leer, **a programar!**
- ¿Dudas?
- ¿Qué tamaño tiene el disco?
Ni idea
- ¿Qué tamaño tiene cada bloque?


(3) El disco

- ¿Qué es? Un montón de bits agrupados en bloques
- A cada bloque lo accedo con su LBA (Logical Block Addressing)
- API de HDD:

```
int read(unsigned int lba, unsigned char * buffer);  
int write(unsigned int lba, unsigned char * buffer);
```
- Con lo que aprendieron y la API para leer, **a programar!**
- ¿Dudas?
- ¿Qué tamaño tiene el disco?
Ni idea
- ¿Qué tamaño tiene cada bloque?
Ni idea

(3) El disco

- ¿Qué es? Un montón de bits agrupados en bloques
- A cada bloque lo accedo con su LBA (Logical Block Addressing)
- API de HDD:

```
int read(unsigned int lba, unsigned char * buffer);  
int write(unsigned int lba, unsigned char * buffer);
```
- Con lo que aprendieron y la API para leer, **a programar!**
- ¿Dudas?
- ¿Qué tamaño tiene el disco?
Ni idea
- ¿Qué tamaño tiene cada bloque?
Ni idea
- ¿Por dónde empiezo? 

(4) MBR

- Master Boot Record

(4) MBR

- Master Boot Record
- El primer bloque del disco

Structure of a classical generic MBR

Address		Description		Size (bytes)
Hex	Dec			
+000h	+0	Bootstrap code area		446
+1BEh	+446	Partition entry #1	Partition table (for primary partitions)	16
+1CEh	+462	Partition entry #2		16
+1DEh	+478	Partition entry #3		16
+1EEh	+494	Partition entry #4		16
+1FEh	+510	55h	Boot signature ^[a]	2
+1FFh	+511	AAh		
Total size: 446 + 4×16 + 2				512

(4) MBR

- Master Boot Record
- El primer bloque del disco

Structure of a classical generic MBR

Address		Description		Size (bytes)
Hex	Dec			
+000h	+0	Bootstrap code area		446
+1BEh	+446	Partition entry #1	Partition table (for primary partitions)	16
+1CEh	+462	Partition entry #2		16
+1DEh	+478	Partition entry #3		16
+1EEh	+494	Partition entry #4		16
+1FEh	+510	55h	Boot signature ^[a]	2
+1FFh	+511	AAh		
Total size: 446 + 4×16 + 2				512

- Esto ya lo tienen resuelto

(5) Partición de EXT2

- Llegamos hasta donde empieza ext2. ¿Y ahora?

(5) Partición de EXT2

- Llegamos hasta donde empieza ext2. ¿Y ahora?
- El superblock: el que tiene **la posta**

(5) Partición de EXT2

- Llegamos hasta donde empieza ext2. ¿Y ahora?
- El superblock: el que tiene **la posta**
- ¿En qué bloque de la partición estará?

(5) Partición de EXT2


- Llegamos hasta donde empieza ext2. ¿Y ahora?
- El superblock: el que tiene **la posta**
- ¿En qué bloque de la partición estará?
- **Exacto, en el tercer bloque**

(5) Partición de EXT2

- Llegamos hasta donde empieza ext2. ¿Y ahora?
- El superblock: el que tiene **la posta**
- ¿En qué bloque de la partición estará?
- Exacto, en el tercer bloque

WTF?

(5) Partición de EXT2

- Llegamos hasta donde empieza ext2. ¿Y ahora?
- El superblock: el que tiene **la posta**
- ¿En qué bloque de la partición estará?
- Exacto, en el tercer bloque
WTF?
- En realidad, siempre en el byte 1024. Independientemente, del tamaño del bloque 

(6) Superblock

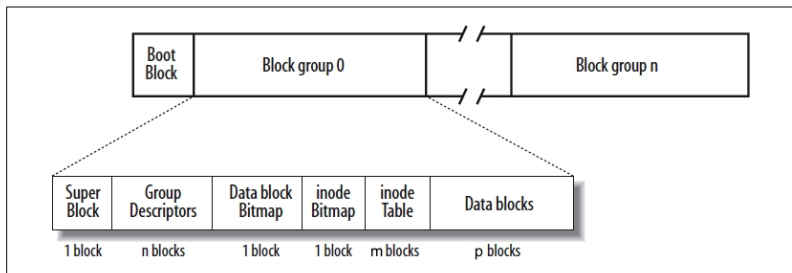
```
struct Ext2FSSuperblock {
__le32  s_inodes_count;           /* Inodes count */
__le32  s_blocks_count;          /* Blocks count */
__le32  s_r_blocks_count;        /* Reserved blocks count */
__le32  s_free_blocks_count;     /* Free blocks count */
__le32  s_free_inodes_count;     /* Free inodes count */
__le32  s_first_data_block;      /* First Data Block */
__le32  s_log_block_size;        /* Block size */
...
__le32  s_blocks_per_group;      /* # Blocks per group */
...
__le32  s_inodes_per_group;      /* # Inodes per group */
...
__le16  s_magic;                 /* Magic signature */
__le32  s_first_ino;             /* First non-reserved inode */
__le16  s_inode_size;            /* size of inode structure
```

(7) Estructura de Ext2

- Todo muy lindo pero ¿dónde están mis archivos?

(7) Estructura de Ext2

- Todo muy lindo pero ¿dónde están mis archivos?



(8) Inodo

- La representación de un archivo

(8) Inodo

- La representación de un archivo
- Un archivo puede ser desde un archivo regular, hasta un directorio, un pipe, un socket, un device, etc.

(8) Inodo


- La representación de un archivo
- Un archivo puede ser desde un archivo regular, hasta un directorio, un pipe, un socket, un device, etc.
- Hoy, para nosotros, una struct de `FSInode`

(9) FSInode

```
struct Ext2FSInode {  
    unsigned short mode;  
    unsigned short uid;  
    unsigned int size;  
    unsigned int atime;  
    unsigned int ctime;  
    unsigned int mtime;  
    unsigned int dtime;  
    unsigned short gid;  
    unsigned short links_count;  
    unsigned int blocks;  
    unsigned int flags;  
    unsigned int os_dependant_1;  
    unsigned int block[15];  
    unsigned int generation;  
    unsigned int file_acl;  
    unsigned int directory_acl;  
    unsigned int faddr;  
    unsigned int os_dependant_2[3];  
}
```



(9) FSInode

```
struct Ext2FSInode {  
    unsigned short mode;  
    unsigned short uid;  
    unsigned int size;  
    unsigned int atime;  
    unsigned int ctime;  
    unsigned int mtime;  
    unsigned int dtime;  
    unsigned short gid;  
    unsigned short links_count;  
    unsigned int blocks;  
    unsigned int flags;  
    unsigned int os_dependant_1;  
    unsigned int block[15];  
    unsigned int generation;  
    unsigned int file_acl;  
    unsigned int directory_acl;  
    unsigned int faddr;  
    unsigned int os_dependant_2[3];  
}
```

- ¿Dónde están los datos? 




(9) FSInode

```
struct Ext2FSInode {  
    unsigned short mode;  
    unsigned short uid;  
    unsigned int size;  
    unsigned int atime;  
    unsigned int ctime;  
    unsigned int mtime;  
    unsigned int dtime;  
    unsigned short gid;  
    unsigned short links_count;  
    unsigned int blocks;  
    unsigned int flags;  
    unsigned int os_dependant_1;  
    unsigned int block[15];  
    unsigned int generation;  
    unsigned int file_acl;  
    unsigned int directory_acl;  
    unsigned int faddr;  
    unsigned int os_dependant_2[3];  
}
```

- ¿Dónde están los datos? 
- ¿Dónde está el nombre del archivo?  Porque la gente no anda preguntando por números de inodos

(9) FSInode

```
struct Ext2FSInode {  
    unsigned short mode;  
    unsigned short uid;  
    unsigned int size;  
    unsigned int atime;  
    unsigned int ctime;  
    unsigned int mtime;  
    unsigned int dtime;  
    unsigned short gid;  
    unsigned short links_count;  
    unsigned int blocks;  
    unsigned int flags;  
    unsigned int os_dependant_1;  
    unsigned int block[15];  
    unsigned int generation;  
    unsigned int file_acl;  
    unsigned int directory_acl;  
    unsigned int faddr;  
    unsigned int os_dependant_2[3];  
}
```

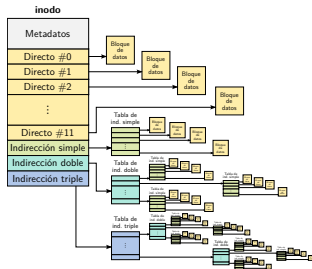
- ¿Dónde están los datos? 
- ¿Dónde está el nombre del archivo?  Porque la gente no anda preguntando por números de inodos
- ¿El inodo directorio qué struct usa? 

(10) Inodo - Datos

- 15 Punteros a bloques con distintos sabores:

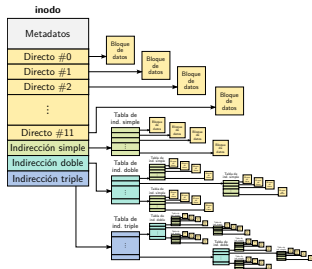
(10) Inodo - Datos

- 15 Punteros a bloques con distintos sabores:
 - 12 Punteros a bloques de datos directos
 - 1 Puntero indirecto a bloque de datos
 - 1 Puntero con una doble indirección a bloque de datos
 - 1 Puntero con una triple indirección a bloque de datos



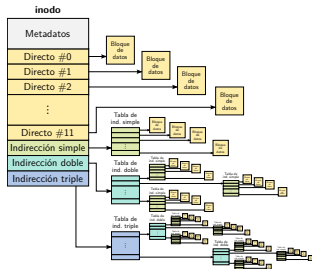
(10) Inodo - Datos

- 15 Punteros a bloques con distintos sabores:
 - 12 Punteros a bloques de datos directos
 - 1 Puntero indirecto a bloque de datos
 - 1 Puntero con una doble indirección a bloque de datos
 - 1 Puntero con una triple indirección a bloque de datos
- ¿Por qué hicieron este quilombo? ⚠



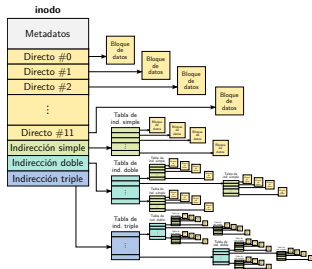
(10) Inodo - Datos

- 15 Punteros a bloques con distintos sabores:
 - 12 Punteros a bloques de datos directos
 - 1 Puntero indirecto a bloque de datos
 - 1 Puntero con una doble indirección a bloque de datos
 - 1 Puntero con una triple indirección a bloque de datos
- ¿Por qué hicieron este quilombo? ⚠
- ¿Son punteros a direcciones de memoria? ⚠



(10) Inodo - Datos

- 15 Punteros a bloques con distintos sabores:
 - 12 Punteros a bloques de datos directos
 - 1 Puntero indirecto a bloque de datos
 - 1 Puntero con una doble indirección a bloque de datos
 - 1 Puntero con una triple indirección a bloque de datos
- ¿Por qué hicieron este quilombo? ⚠
- ¿Son punteros a direcciones de memoria? ⚠
- ¿Los bloques a los que apuntan, están en memoria o en disco? ⚠



(11) Inodo - Directorio

- Es un inodo IGUAL que cualquier otro

(11) Inodo - Directorio

- Es un inodo IGUAL que cualquier otro
- Es decir, tiene la misma estructura Ext2FSInode


(11) Inodo - Directorio

- Es un inodo IGUAL que cualquier otro
- Es decir, tiene la misma estructura Ext2FSInode
- Entonces ¿Dónde están los archivos de mi directorio?

(11) Inodo - Directorio

- Es un inodo IGUAL que cualquier otro
- Es decir, tiene la misma estructura Ext2FSInode
- Entonces ¿Dónde están los archivos de mi directorio?
- En los bloques de datos

(11) Inodo - Directorio

- Es un inodo IGUAL que cualquier otro
- Es decir, tiene la misma estructura Ext2FSInode
- Entonces ¿Dónde están los archivos de mi directorio?
- En los bloques de datos
- Repito, en los bloques de datos 


(12) Ext2FSInode

```
struct Ext2FSDirEntry {  
    unsigned int inode;  
    unsigned short record_length;  
    unsigned char name_length;  
    unsigned char file_type;  
    char name[];  
};
```

- **Los datos** del inodo son un arreglo de struct Ext2FSDirEntry


(12) Ext2FSInode

```
struct Ext2FSDirEntry {  
    unsigned int inode;  
    unsigned short record_length;  
    unsigned char name_length;  
    unsigned char file_type;  
    char name[];  
};
```

- **Los datos** del inodo son un arreglo de struct Ext2FSDirEntry
- La struct tiene tamaño variable 


(12) Ext2FSInode

```
struct Ext2FSDirEntry {  
    unsigned int inode;  
    unsigned short record_length;  
    unsigned char name_length;  
    unsigned char file_type;  
    char name[];  
};
```

- **Los datos** del inodo son un arreglo de struct Ext2FSDirEntry
- La struct tiene tamaño variable 
- ¿Cómo saber cuantas structs tengo en mi arreglo? Para pensar

(12) Ext2FSInode

```
struct Ext2FSDirEntry {  
    unsigned int inode;  
    unsigned short record_length;  
    unsigned char name_length;  
    unsigned char file_type;  
    char name[];  
};
```

- **Los datos** del inodo son un arreglo de struct Ext2FSDirEntry
- La struct tiene tamaño variable 
- ¿Cómo saber cuantas structs tengo en mi arreglo? Para pensar
- ¿De verdad vas a usar un arreglo si tienen tamaño variable?
Apa-la-la

(13) Enunciado

- Completar la implementación de los siguientes métodos:

(13) Enunciado

- Completar la implementación de los siguientes métodos:
 - ① `unsigned int get_block_address(inode,block_number)`

(13) Enunciado

- Completar la implementación de los siguientes métodos:
 - ① `unsigned int get_block_address(inode, block_number)`
 - Dado un inodo, devuelve la dirección (el lba) de su i-ésimo bloque (sólo hasta primera indirección)

(13) Enunciado

- Completar la implementación de los siguientes métodos:
 - ❶ `unsigned int get_block_address(inode,block_number)`
 - Dado un inodo, devuelve la dirección (el lba) de su i-ésimo bloque (sólo hasta primera indirección)
 - ❷ `Ext2FSInode * load_inode(inode_number)`

(13) Enunciado

- Completar la implementación de los siguientes métodos:
 - ❶ `unsigned int get_block_address(inode, block_number)`
 - Dado un inodo, devuelve la dirección (el lba) de su i-ésimo bloque (sólo hasta primera indirección)
 - ❷ `Ext2FSInode * load_inode(inode_number)`
 - Obtiene el i-ésimo inodo y lo carga en memoria

(13) Enunciado

- Completar la implementación de los siguientes métodos:
 - ➊ `unsigned int get_block_address(inode,block_number)`
 - Dado un inodo, devuelve la dirección (el lba) de su i-ésimo bloque (sólo hasta primera indirección)
 - ➋ `Ext2FSInode * load_inode(inode_number)`
 - Obtiene el i-ésimo inodo y lo carga en memoria
 - ➌ `Ext2FSInode * get_file_inode_from_dir_inode(from,filename)`

(13) Enunciado

- Completar la implementación de los siguientes métodos:
 - ❶ `unsigned int get_block_address(inode,block_number)`
 - Dado un inodo, devuelve la dirección (el lba) de su i-ésimo bloque (sólo hasta primera indirección)
 - ❷ `Ext2FSInode * load_inode(inode_number)`
 - Obtiene el i-ésimo inodo y lo carga en memoria
 - ❸ `Ext2FSInode * get_file_inode_from_dir_inode(from,filename)`
 - Dado un inodo directorio, obtiene el inodo de un archivo contenido en ese directorio, a través del nombre


(14) ¿Qué tengo solucionado?

- Clases HDD, MBR y PartitionEntry resueltas


(14) ¿Qué tengo solucionado?

- Clases HDD, MBR y PartitionEntry resueltas
- Clase Ext2FS parcialmente


(14) ¿Qué tengo solucionado?

- Clases HDD, MBR y PartitionEntry resueltas
- Clase Ext2FS parcialmente
- Estructuras de Ext2FS 


(14) ¿Qué tengo solucionado?

- Clases HDD, MBR y PartitionEntry resueltas
- Clase Ext2FS parcialmente
- Estructuras de Ext2FS 
 - Ext2FSSuperblock (Superblock)


(14) ¿Qué tengo solucionado?

- Clases HDD, MBR y PartitionEntry resueltas
- Clase Ext2FS parcialmente
- Estructuras de Ext2FS 
 - Ext2FSSuperblock (Superblock)
 - Ext2FSBlockGroupDescriptor (Block Group Descriptor)

(14) ¿Qué tengo solucionado?

- Clases HDD, MBR y PartitionEntry resueltas
- Clase Ext2FS parcialmente
- Estructuras de Ext2FS 
 - Ext2FSSuperblock (Superblock)
 - Ext2FSBlockGroupDescriptor (Block Group Descriptor)
 - Ext2FSInode (Inode)



(14) ¿Qué tengo solucionado?

- Clases HDD, MBR y PartitionEntry resueltas
- Clase Ext2FS parcialmente
- Estructuras de Ext2FS 
 - Ext2FSSuperblock (Superblock)
 - Ext2FSBlockGroupDescriptor (Block Group Descriptor)
 - Ext2FSInode (Inode)
 - Ext2FSDirEntry (Directory Entry)

(14) ¿Qué tengo solucionado?

- Clases HDD, MBR y PartitionEntry resueltas
- Clase Ext2FS parcialmente
- Estructuras de Ext2FS ⚠
 - Ext2FSSuperblock (Superblock)
 - Ext2FSBlockGroupDescriptor (Block Group Descriptor)
 - Ext2FSInode (Inode)
 - Ext2FSDirEntry (Directory Entry)
- Funciones auxiliares de Ext2FS ⚠

(14) ¿Qué tengo solucionado?

- Clases HDD, MBR y PartitionEntry resueltas
- Clase Ext2FS parcialmente
- Estructuras de Ext2FS 
 - Ext2FSSuperblock (Superblock)
 - Ext2FSBlockGroupDescriptor (Block Group Descriptor)
 - Ext2FSInode (Inode)
 - Ext2FSDirEntry (Directory Entry)
- Funciones auxiliares de Ext2FS 
 - `read_block`: Lee un bloque de disco



(14) ¿Qué tengo solucionado?

- Clases HDD, MBR y PartitionEntry resueltas
- Clase Ext2FS parcialmente
- Estructuras de Ext2FS 
 - Ext2FSSuperblock (Superblock)
 - Ext2FSBlockGroupDescriptor (Block Group Descriptor)
 - Ext2FSInode (Inode)
 - Ext2FSDirEntry (Directory Entry)
- Funciones auxiliares de Ext2FS 
 - read_block: Lee un bloque de disco
 - superblock: Devuelve el superbloque



(14) ¿Qué tengo solucionado?

- Clases HDD, MBR y PartitionEntry resueltas
- Clase Ext2FS parcialmente
- Estructuras de Ext2FS ⚠
 - Ext2FSSuperblock (Superblock)
 - Ext2FSBlockGroupDescriptor (Block Group Descriptor)
 - Ext2FSInode (Inode)
 - Ext2FSDirEntry (Directory Entry)
- Funciones auxiliares de Ext2FS ⚠
 - read_block: Lee un bloque de disco
 - superblock: Devuelve el superbloque
 - block_group: Devuelve el descriptor del bloque de grupo

(14) ¿Qué tengo solucionado?

- Clases HDD, MBR y PartitionEntry resueltas
- Clase Ext2FS parcialmente
- Estructuras de Ext2FS 
 - Ext2FSSuperblock (Superblock)
 - Ext2FSBlockGroupDescriptor (Block Group Descriptor)
 - Ext2FSInode (Inode)
 - Ext2FSDirEntry (Directory Entry)
- Funciones auxiliares de Ext2FS 
 - read_block: Lee un bloque de disco
 - superblock: Devuelve el superbloque
 - block_group: Devuelve el descriptor del bloque de grupo
 - blockgroup_for_inode: Número de blockgroup del inodo

(14) ¿Qué tengo solucionado?

- Clases HDD, MBR y PartitionEntry resueltas
- Clase Ext2FS parcialmente
- Estructuras de Ext2FS 
 - Ext2FSSuperblock (Superblock)
 - Ext2FSBlockGroupDescriptor (Block Group Descriptor)
 - Ext2FSInode (Inode)
 - Ext2FSDirEntry (Directory Entry)
- Funciones auxiliares de Ext2FS 
 - read_block: Lee un bloque de disco
 - superblock: Devuelve el superbloque
 - block_group: Devuelve el descriptor del bloque de grupo
 - blockgroup_for_inode: Número de blockgroup del inodo
 - blockgroup_inode_index: Offset dentro de la tabla de inodos para el inodo

(15) Últimos tips

- Hagan los ejercicios en el orden dado

(15) Últimos tips

- Hagan los ejercicios en el orden dado
- Descarguen los archivos en /tmp


(15) Últimos tips

- Hagan los ejercicios en el orden dado
- Descarguen los archivos en /tmp
- Hay estructuras para cada tipo


(15) Últimos tips

- Hagan los ejercicios en el orden dado
- Descarguen los archivos en /tmp
- Hay estructuras para cada tipo
- Utilicen las funciones auxiliares

(15) Últimos tips

- Hagan los ejercicios en el orden dado
- Descarguen los archivos en /tmp
- Hay estructuras para cada tipo
- Utilicen las funciones auxiliares
- Los directorios son archivos 

(15) Últimos tips

- Hagan los ejercicios en el orden dado
- Descarguen los archivos en /tmp
- Hay estructuras para cada tipo
- Utilicen las funciones auxiliares
- Los directorios son archivos 
- ¿Los directorios son archivos?

(15) Últimos tips

- Hagan los ejercicios en el orden dado
- Descarguen los archivos en /tmp
- Hay estructuras para cada tipo
- Utilicen las funciones auxiliares
- Los directorios son archivos ⚠
- ¿Los directorios son archivos?
- Sí, los directorios son archivos ⚠



(15) Últimos tips

- Hagan los ejercicios en el orden dado
- Descarguen los archivos en /tmp
- Hay estructuras para cada tipo
- Utilicen las funciones auxiliares
- Los directorios son archivos ⚠
- ¿Los directorios son archivos?
- Sí, los directorios son archivos ⚠
- Documentación



(15) Últimos tips

- Hagan los ejercicios en el orden dado
- Descarguen los archivos en /tmp
- Hay estructuras para cada tipo
- Utilicen las funciones auxiliares
- Los directorios son archivos ⚠
- ¿Los directorios son archivos?
- Sí, los directorios son archivos ⚠
- Documentación
 - <http://www.nongnu.org/ext2-doc/ext2.html>

(15) Últimos tips

- Hagan los ejercicios en el orden dado
- Descarguen los archivos en /tmp
- Hay estructuras para cada tipo
- Utilicen las funciones auxiliares
- Los directorios son archivos 
- ¿Los directorios son archivos?
- Sí, los directorios son archivos 
- Documentación
 - <http://www.nongnu.org/ext2-doc/ext2.html>
 - <http://e2fsprogs.sourceforge.net/ext2intro.html>

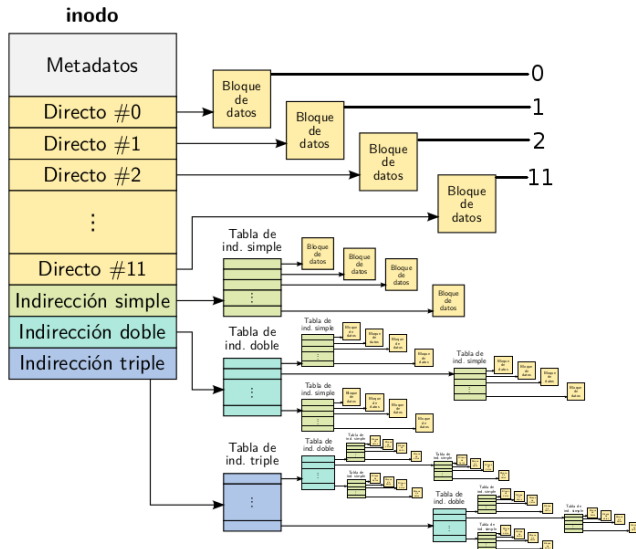
(15) Últimos tips

- Hagan los ejercicios en el orden dado
- Descarguen los archivos en /tmp
- Hay estructuras para cada tipo
- Utilicen las funciones auxiliares
- Los directorios son archivos 
- ¿Los directorios son archivos?
- Sí, los directorios son archivos 
- Documentación
 - <http://www.nongnu.org/ext2-doc/ext2.html>
 - <http://e2fsprogs.sourceforge.net/ext2intro.html>
 - <http://wiki.osdev.org/Ext2>

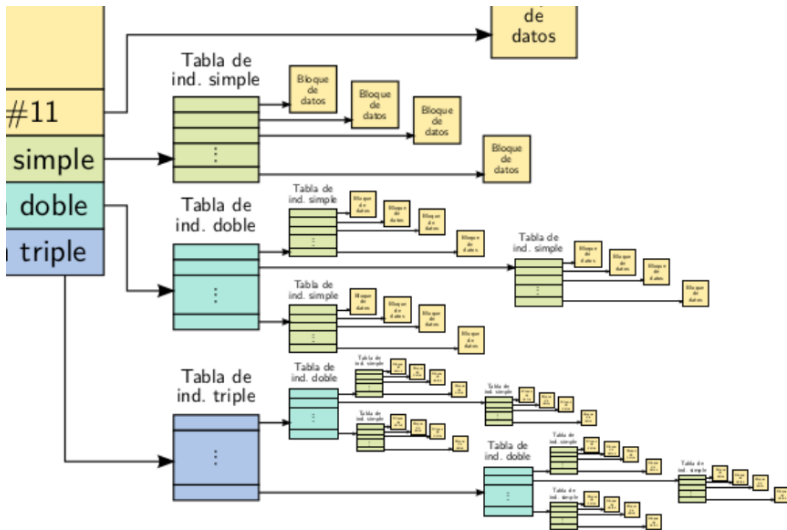
(15) Últimos tips

- Hagan los ejercicios en el orden dado
- Descarguen los archivos en /tmp
- Hay estructuras para cada tipo
- Utilicen las funciones auxiliares
- Los directorios son archivos ⚠
- ¿Los directorios son archivos?
- Sí, los directorios son archivos ⚠
- Documentación
 - <http://www.nongnu.org/ext2-doc/ext2.html>
 - <http://e2fsprogs.sourceforge.net/ext2intro.html>
 - <http://wiki.osdev.org/Ext2>
 - <http://oreilly.com/catalog/linuxkernel2/chapter/ch17.pdf>

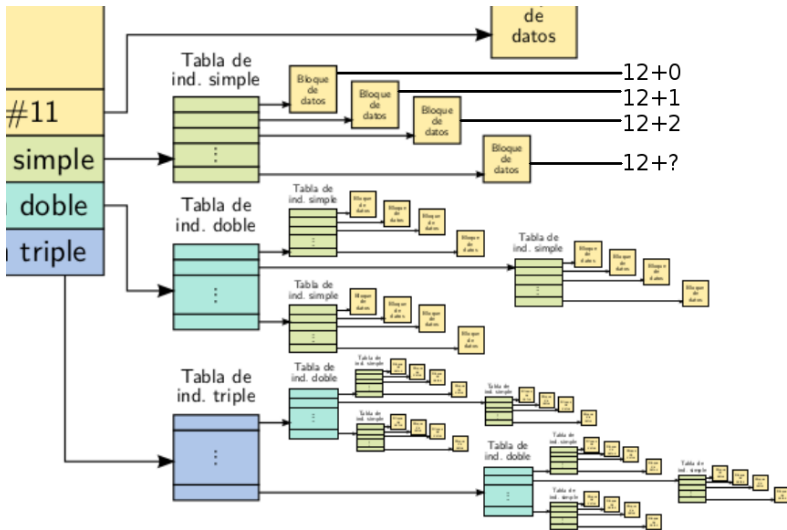
(17) get_block_address



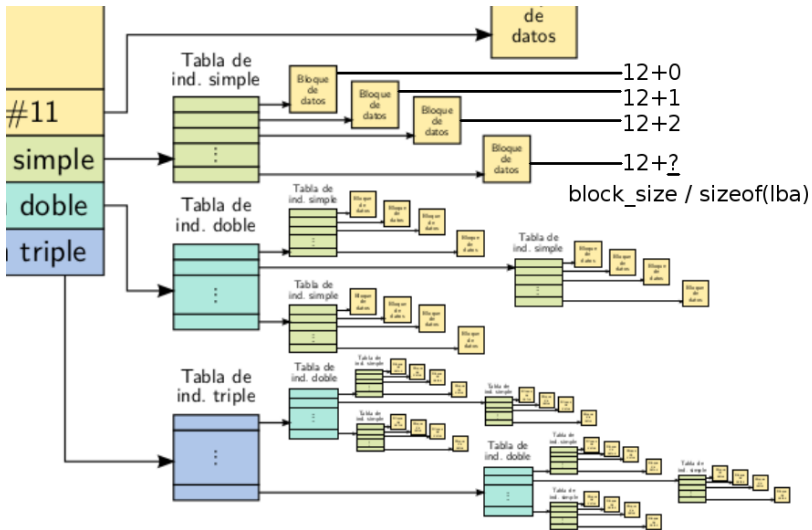
(18) get_block_address



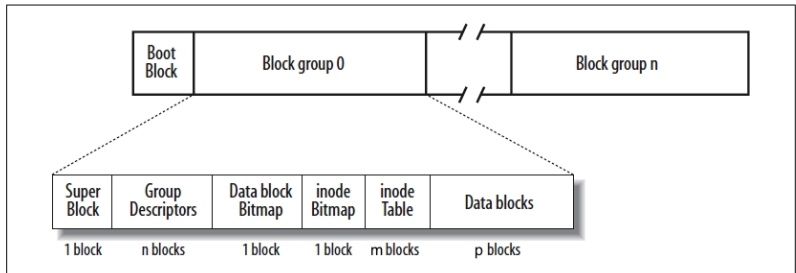
(19) get_block_address



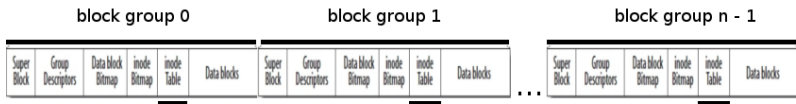
(20) get_block_address



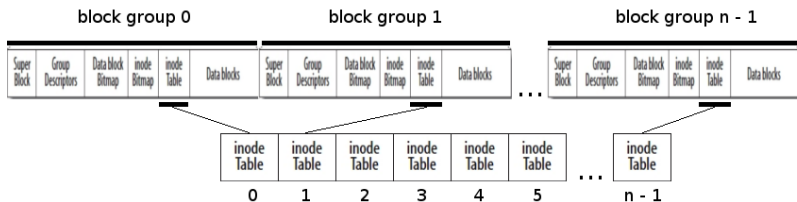
(21) load_inode



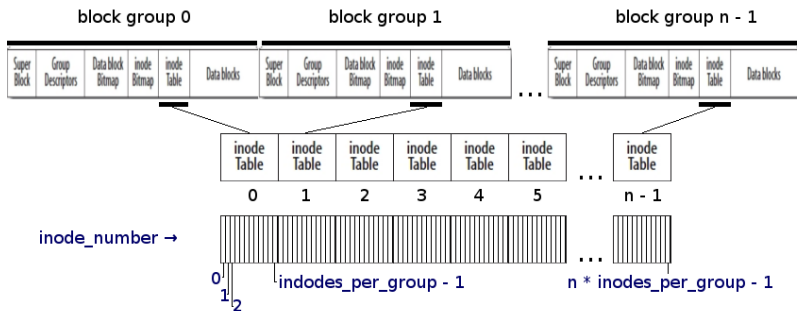
(22) load_inode



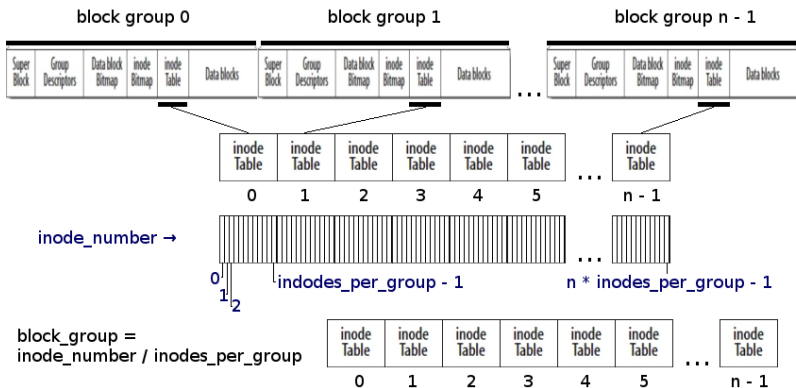
(23) load_inode



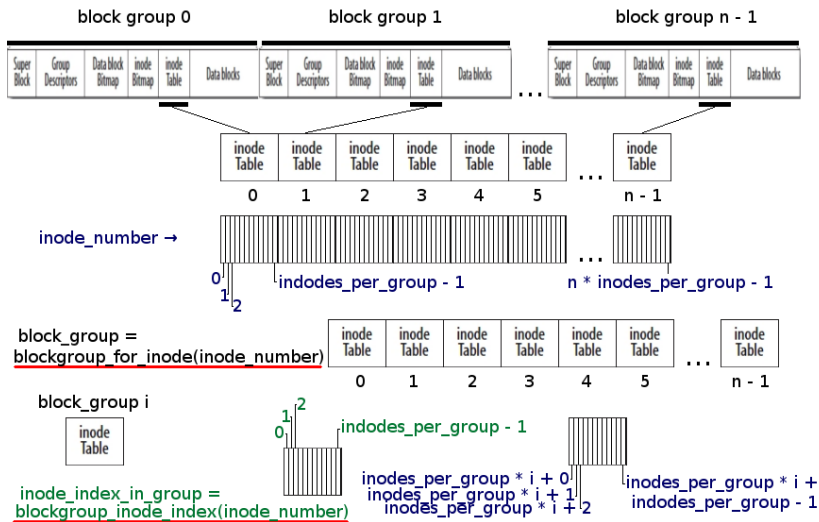
(24) load_inode



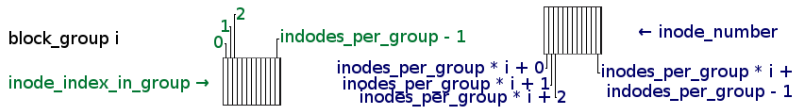
(25) load_inode



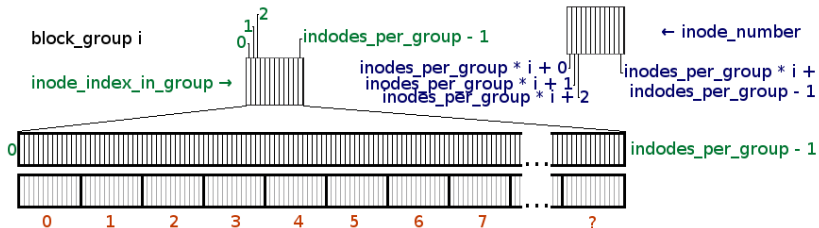
(27) load_inode



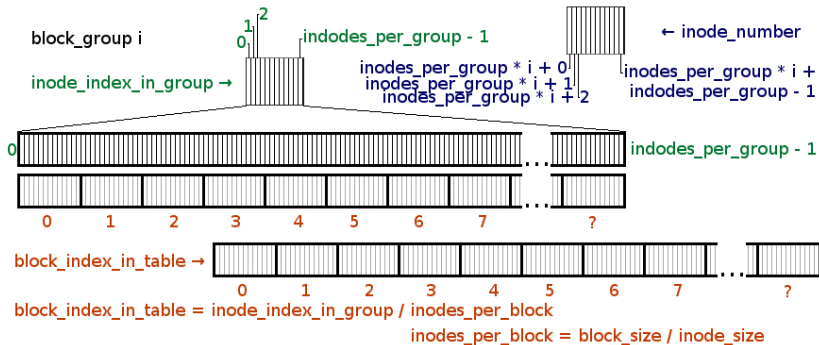
(28) load_inode



(29) load_inode



(30) load_inode



(32) load_inode

```
¿read_block(block_index_in_table)?
```

(33) load_inode

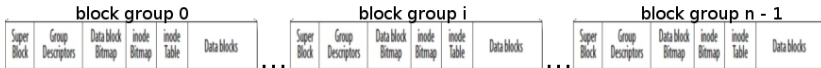
¿read_block(block_index_in_table)?

¡NO! read_block(block_index)

(34) load_inode

¿read_block(block_index_in_table)?

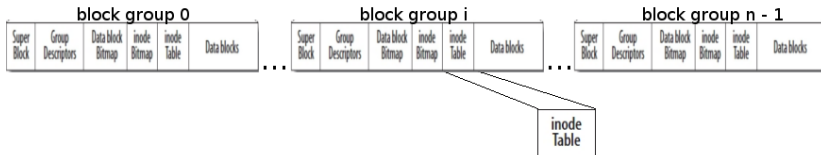
¡NO! read_block(block_index)



(35) load_inode

¿read_block(block_index_in_table)?

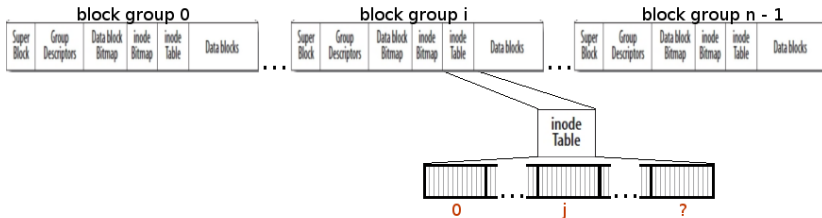
¡NO! read_block(block_index)



(36) load_inode

¿read_block(block_index_in_table)?

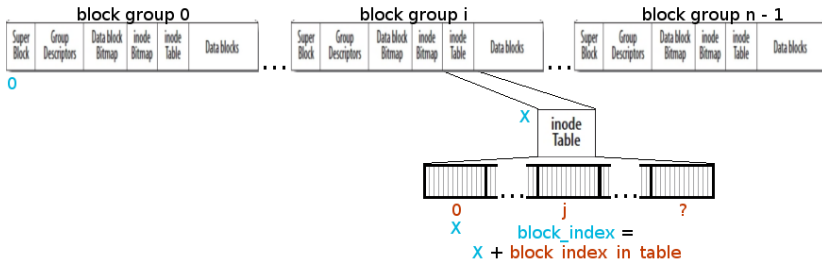
¡NO! read_block(block_index)



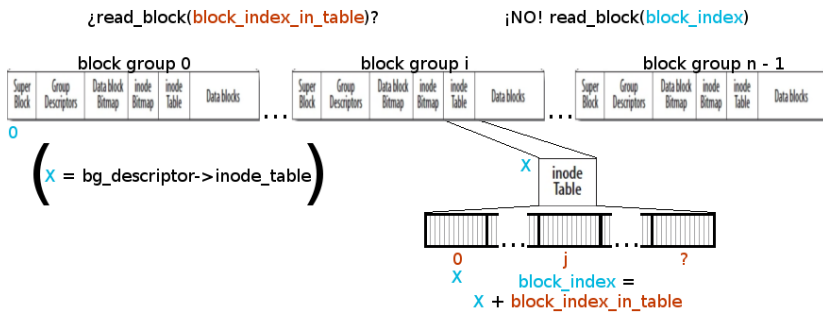
(37) load_inode

¿read_block(block_index_in_table)?

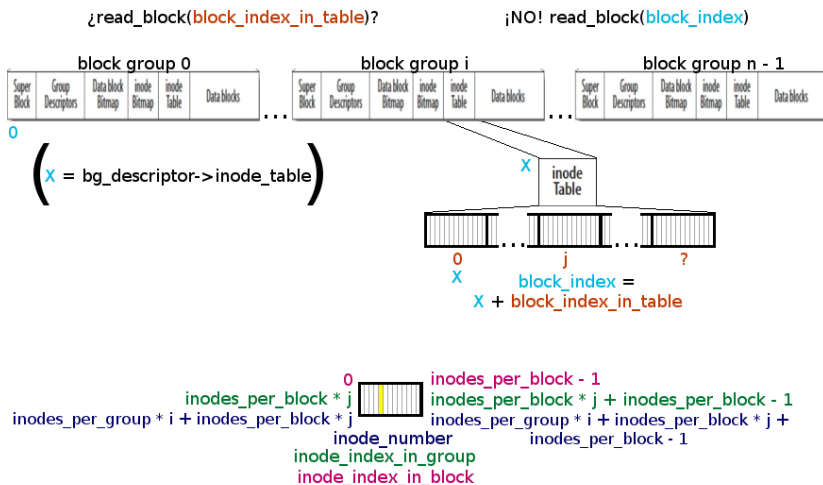
¡NO! read_block(block_index)



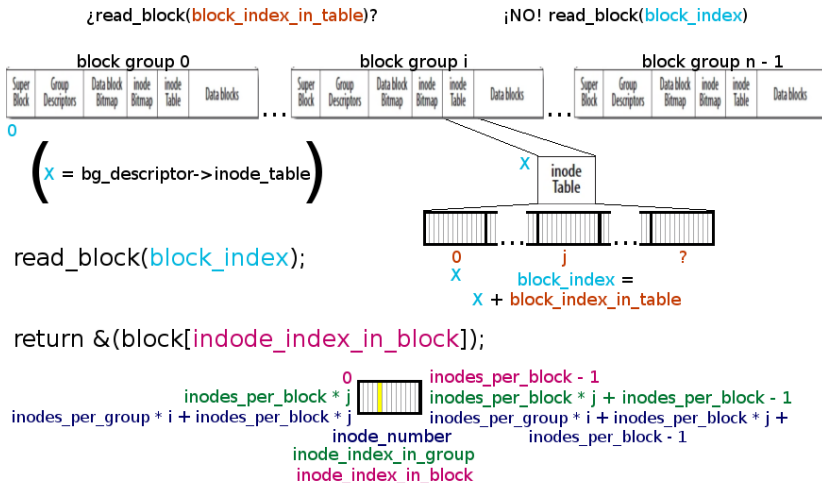
(38) load_inode



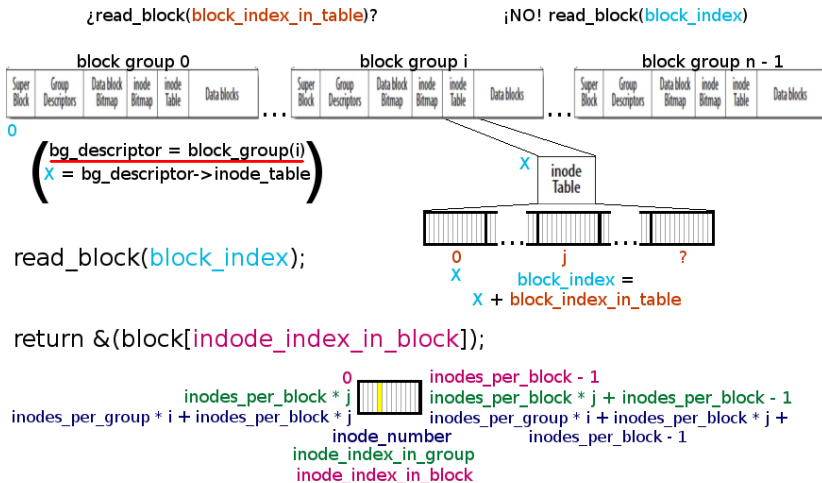
(39) load_inode



(40) load_inode



(41) load_inode



(42) get_file_inode_from_dir_inode

	inode	rec_len	file_type	name_len	name	
0	13	12	1	2	. \0 \0 \0	.
12	2	12	2	2	. . \0 \0	..
24	18	16	5	2	m u s i c \0 \0 \0	music/
40	15	16	8	1	t e s t . t x t	test.txt
56	19	12	3	2	b i n \0	bin/