Архитектура операционной системы

Files, filesystems, vfs

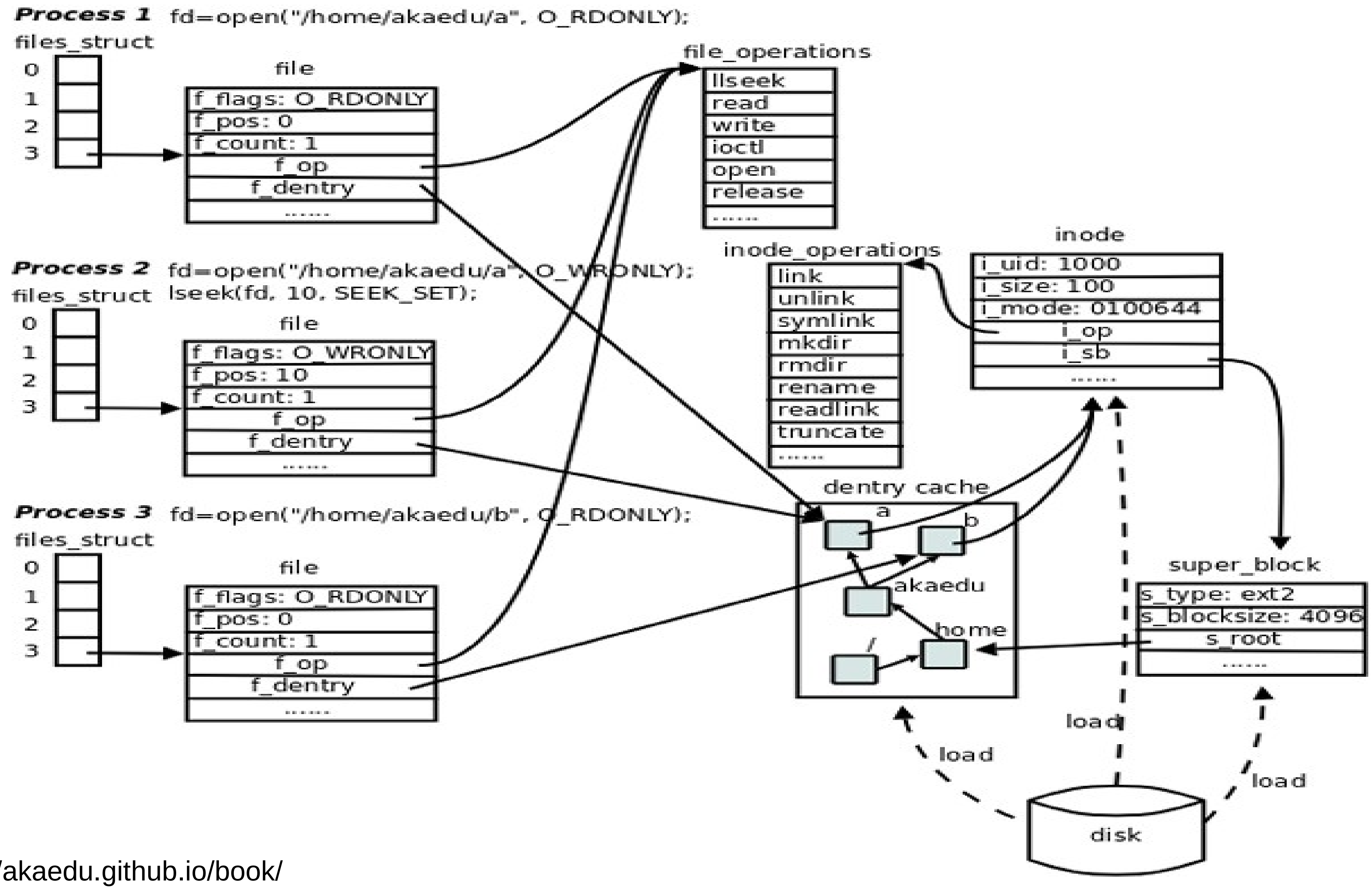
Назначение

- Реализация единого интерфейса для построения файловых систем
- Уровень абстракции от конкретной файловой системы
- Стандартизация доступа к ФС на разных носителях

Структуры данных VFS

- superblock – главный (первый) блок файловой системы, описывающий ее параметры (метаинформация о ФС)
- inode – индексный узел (метаинформация о файле)
- dentry – элемент пути (каталог)
- file – объект, представляющий собой открытый файл, связанный с процессом

Связь структур данных



http://akaedu.github.io/book/

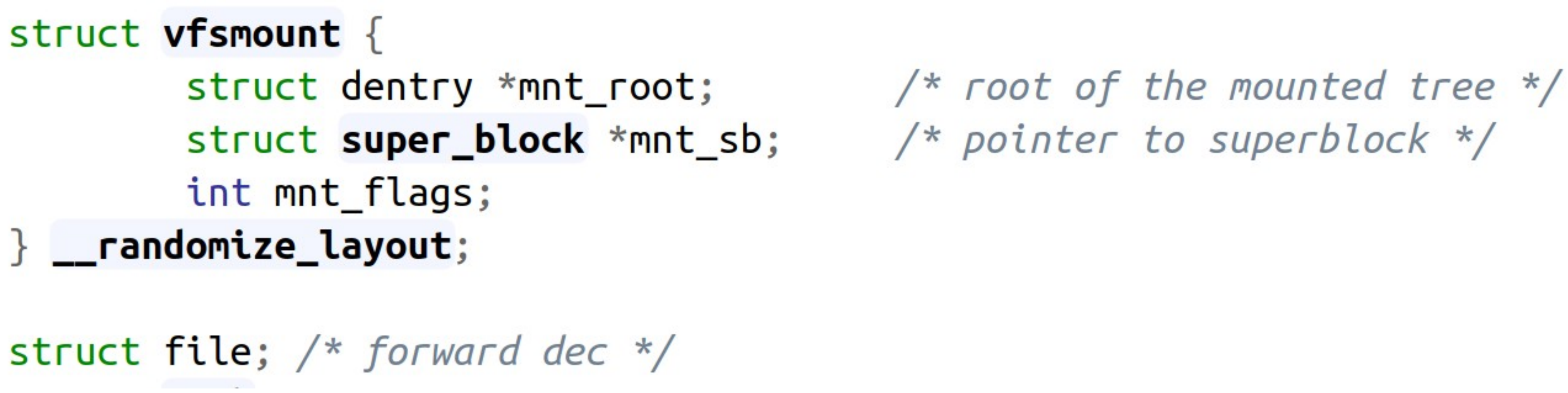
file_systems→* file_system_type

```
struct file_system_type {
        const char *name;
        int fs_flags;
#define FS_REQUIRES_DEV         1
#define FS_BINARY_MOUNTDATA     2
#define FS_HAS_SUBTYPE          4
#define FS_USERNS_MOUNT         8       /* Can be mounted by userns root */
#define FS_RENAME_DOES_D_MOVE   32768   /* FS will handle d_move() during rename() internally. */
        struct dentry *(*mount) (struct file_system_type *, int,
                       const char *, void *);
        void (*kill_sb) (struct super_block *);
        struct module *owner;
        struct file_system_type * next;
        struct hlist_head fs_supers;

        struct lock_class_key s_lock_key;
        struct lock_class_key s_umount_key;
        struct lock_class_key s_vfs_rename_key;
        struct lock_class_key s_writers_key[SB_FREEZE_LEVELS];

        struct lock_class_key i_lock_key;
        struct lock_class_key i_mutex_key;
        struct lock_class_key i_mutex_dir_key;
};
```

Структуры данных VFS

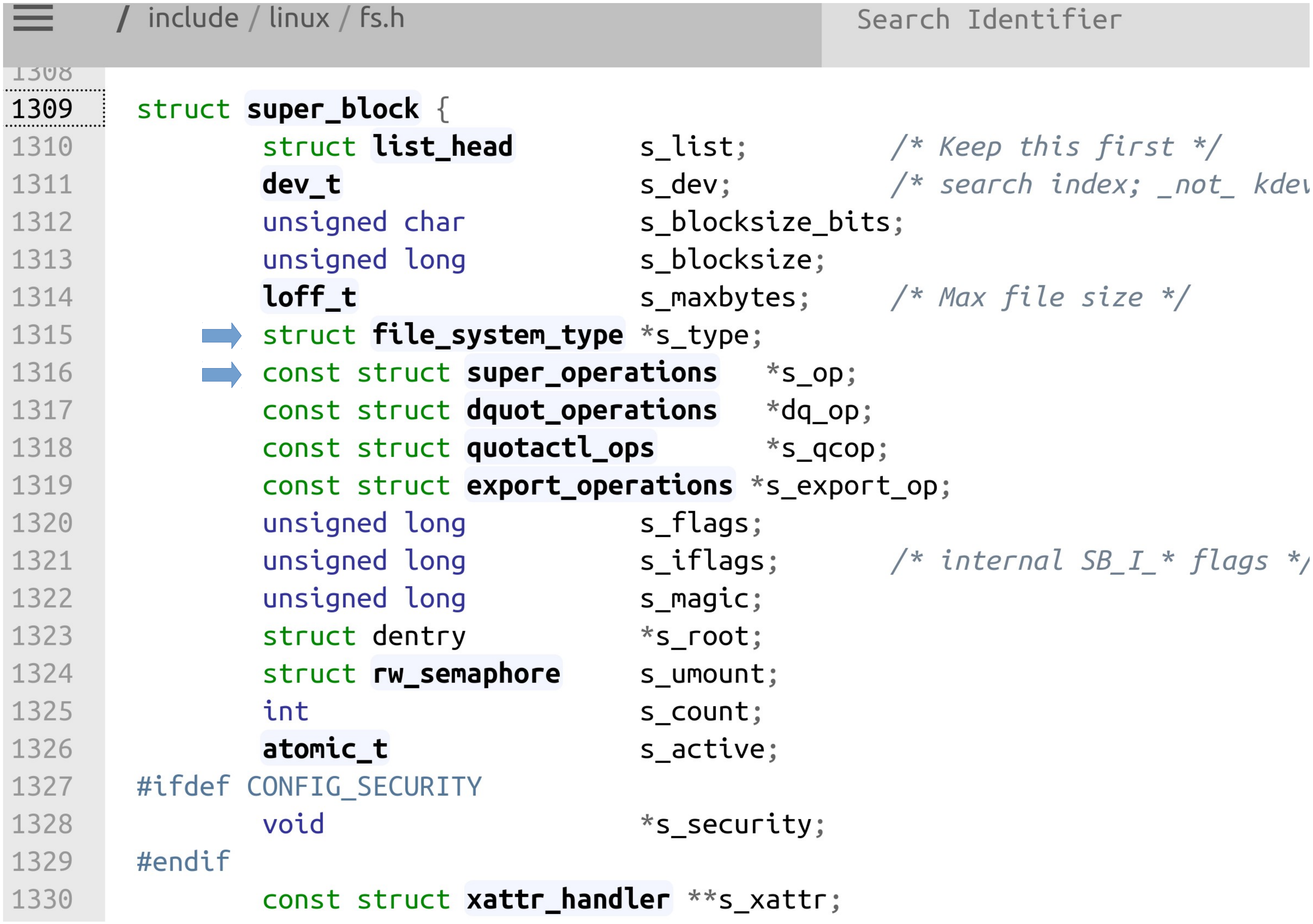
- superblock – главный (первый) блок файловой системы, описывающий ее параметры (метаинформация о ФС)
- inode – индексный узел (метаинформация о файле)
- dentry – элемент пути (каталог)
- file – объект, представляющий собой открытый файл, связанный с процессом

current→namespace→* vfsmount

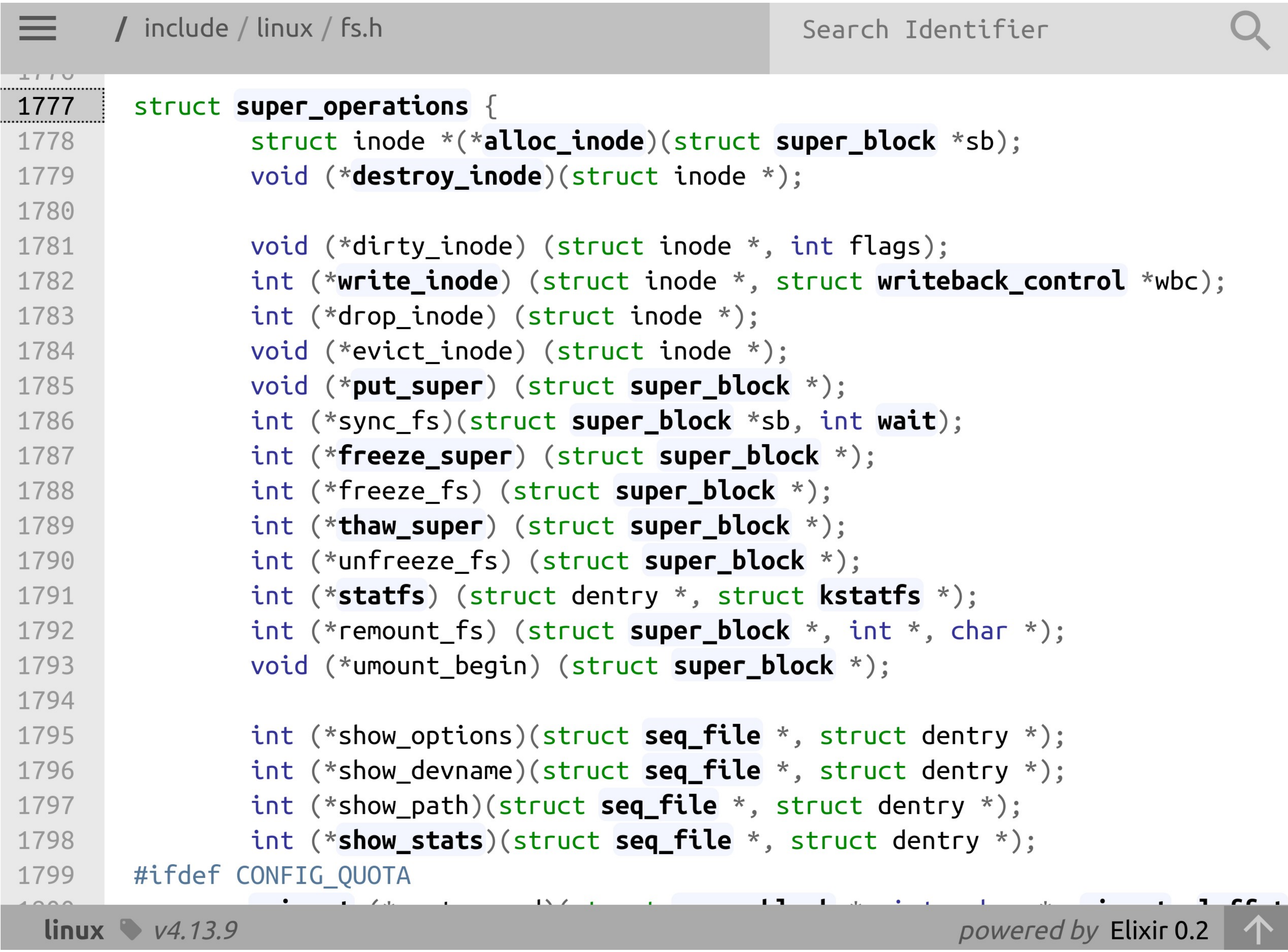


```
struct vfsmount {
        struct dentry *mnt_root;        /* root of the mounted tree */
        struct super_block *mnt_sb;     /* pointer to superblock */
        int mnt_flags;
} __randomize_layout;

struct file; /* forward dec */
```

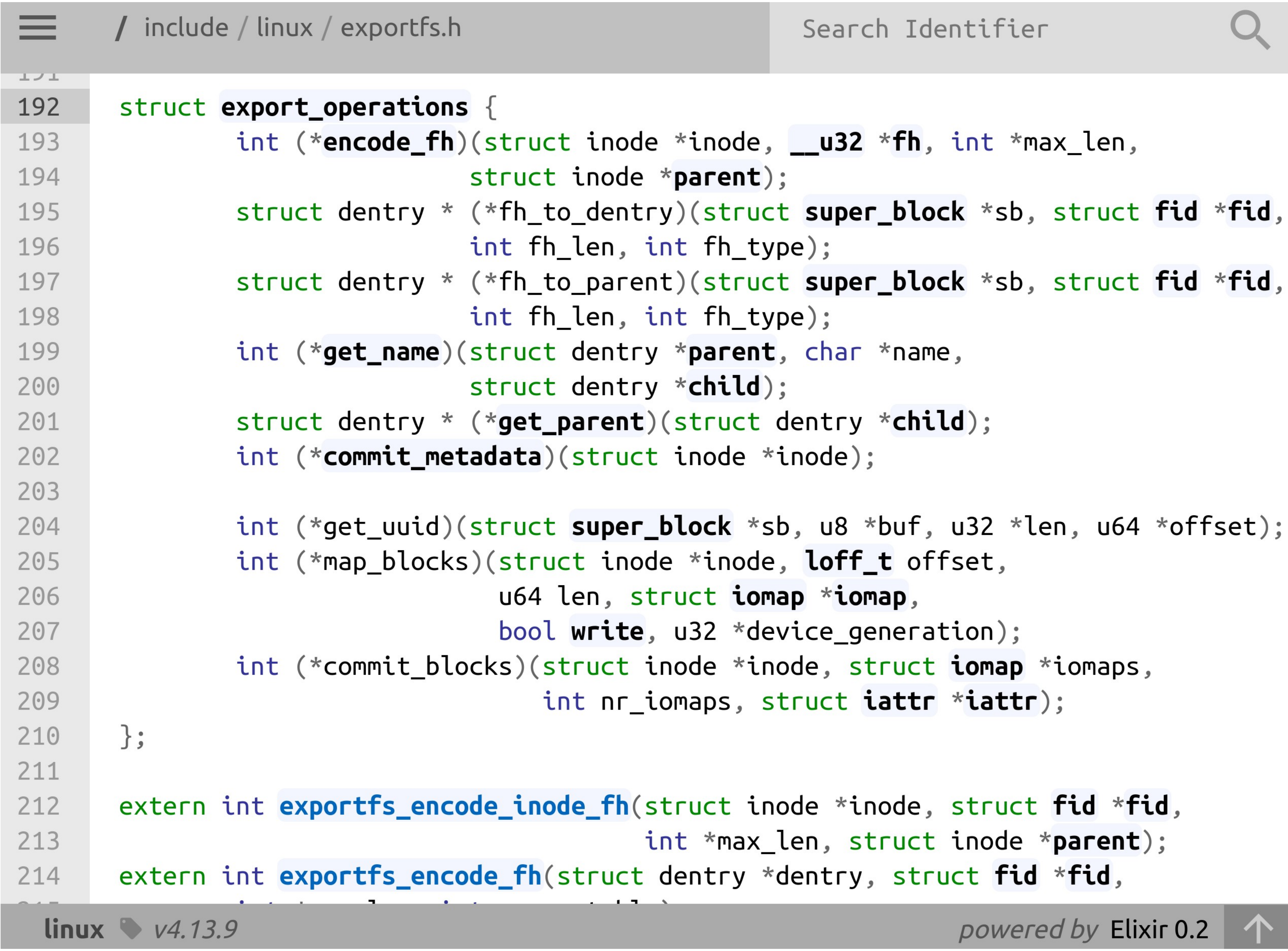
Current→namespace→* **mnt_sb**



```
struct super_block {
        struct list_head        s_list;         /* Keep this first */
        dev_t                   s_dev;          /* search index; _not_ kdev
        unsigned char           s_blocksize_bits;
        unsigned long           s_blocksize;
        loff_t                  s_maxbytes;     /* Max file size */
        struct file_system_type *s_type;
        const struct super_operations   *s_op;
        const struct dquot_operations   *dq_op;
        const struct quotactl_ops       *s_qcop;
        const struct export_operations *s_export_op;
        unsigned long           s_flags;
        unsigned long           s_iflags;       /* internal SB_I_* flags */
        unsigned long           s_magic;
        struct dentry           *s_root;
        struct rw_semaphore     s_umount;
        int                     s_count;
        atomic_t                s_active;
#ifdef CONFIG_SECURITY
        void                    *s_security;
#endif
        const struct xattr_handler **s_xattr;
```



```c
struct super_operations {
        struct inode *(*alloc_inode)(struct super_block *sb);
        void (*destroy_inode)(struct inode *);

        void (*dirty_inode) (struct inode *, int flags);
        int (*write_inode) (struct inode *, struct writeback_control *wbc);
        int (*drop_inode) (struct inode *);
        void (*evict_inode) (struct inode *);
        void (*put_super) (struct super_block *);
        int (*sync_fs)(struct super_block *sb, int wait);
        int (*freeze_super) (struct super_block *);
        int (*freeze_fs) (struct super_block *);
        int (*thaw_super) (struct super_block *);
        int (*unfreeze_fs) (struct super_block *);
        int (*statfs) (struct dentry *, struct kstatfs *);
        int (*remount_fs) (struct super_block *, int *, char *);
        void (*umount_begin) (struct super_block *);

        int (*show_options)(struct seq_file *, struct dentry *);
        int (*show_devname)(struct seq_file *, struct dentry *);
        int (*show_path)(struct seq_file *, struct dentry *);
        int (*show_stats)(struct seq_file *, struct dentry *);
#ifdef CONFIG_QUOTA
```

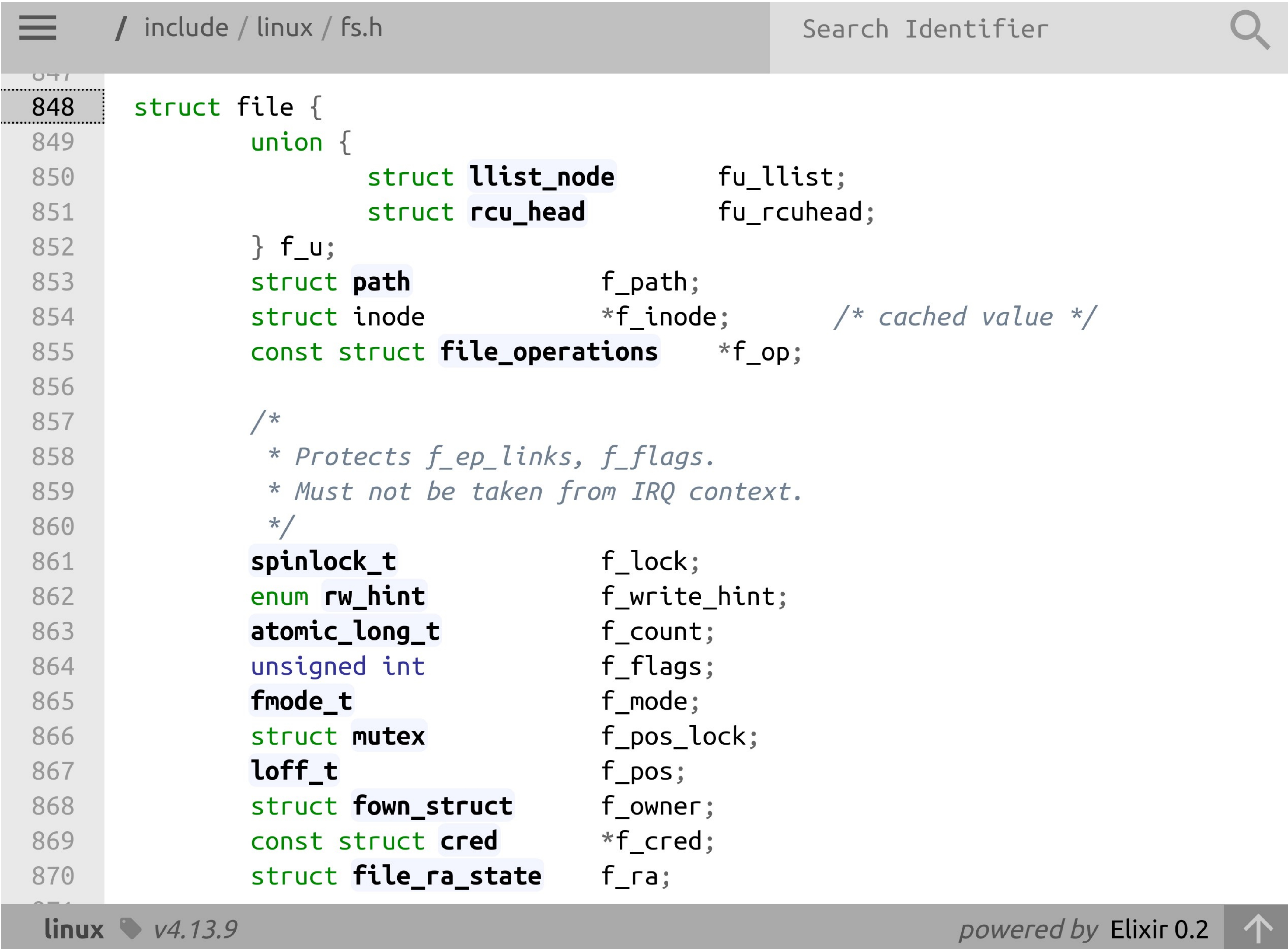


/ include / linux / exportfs.h

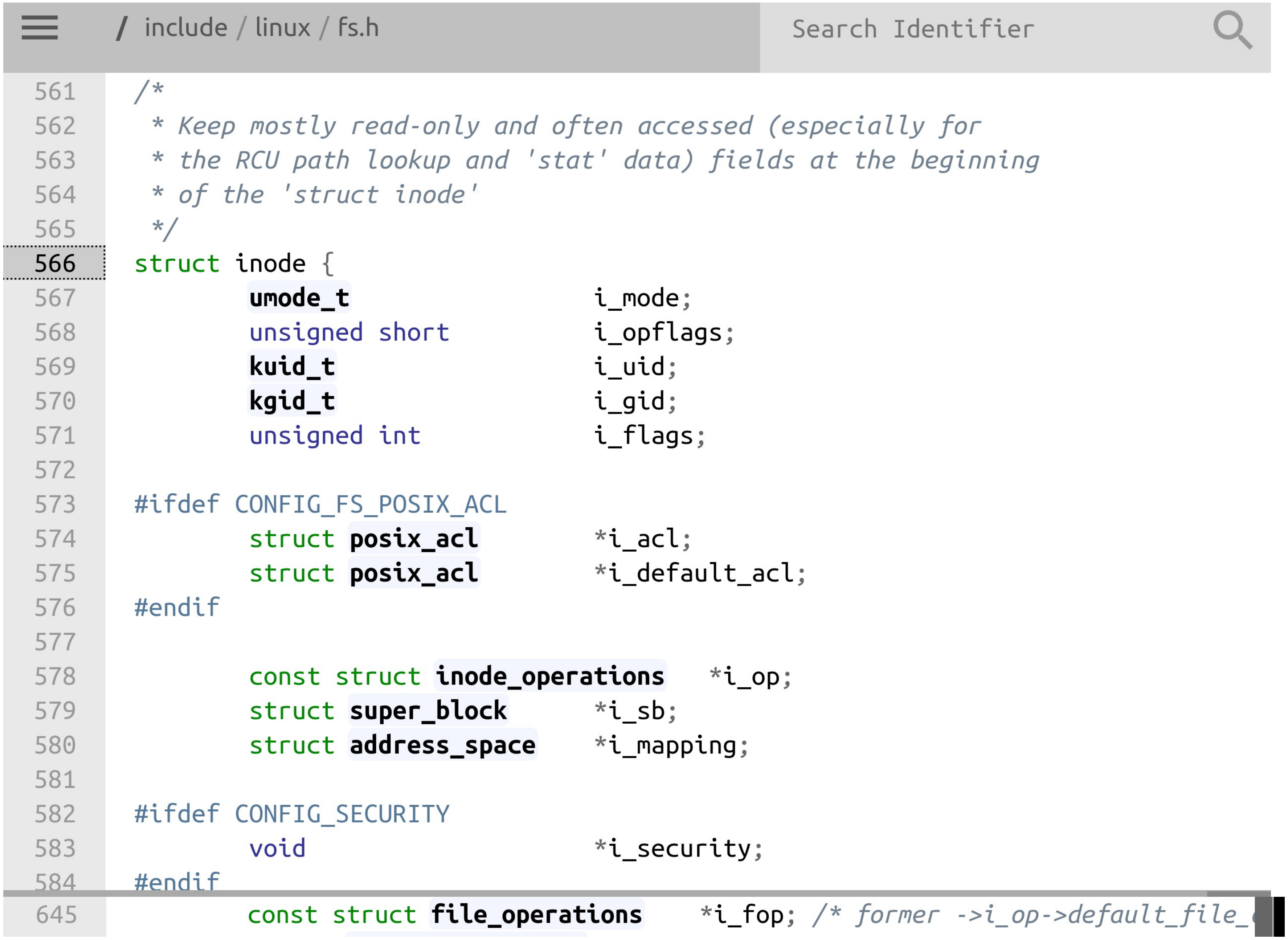
Search Identifier

```c
struct export_operations {
        int (*encode_fh)(struct inode *inode, __u32 *fh, int *max_len,
                        struct inode *parent);
        struct dentry * (*fh_to_dentry)(struct super_block *sb, struct fid *fid,
                        int fh_len, int fh_type);
        struct dentry * (*fh_to_parent)(struct super_block *sb, struct fid *fid,
                        int fh_len, int fh_type);
        int (*get_name)(struct dentry *parent, char *name,
                        struct dentry *child);
        struct dentry * (*get_parent)(struct dentry *child);
        int (*commit_metadata)(struct inode *inode);

        int (*get_uuid)(struct super_block *sb, u8 *buf, u32 *len, u64 *offset);
        int (*map_blocks)(struct inode *inode, loff_t offset,
                          u64 len, struct iomap *iomap,
                          bool write, u32 *device_generation);
        int (*commit_blocks)(struct inode *inode, struct iomap *iomaps,
                             int nr_iomaps, struct iattr *iattr);
};

extern int exportfs_encode_inode_fh(struct inode *inode, struct fid *fid,
                                    int *max_len, struct inode *parent);
extern int exportfs_encode_fh(struct dentry *dentry, struct fid *fid,
```

linux v4.13.9 *powered by* Elixir 0.2

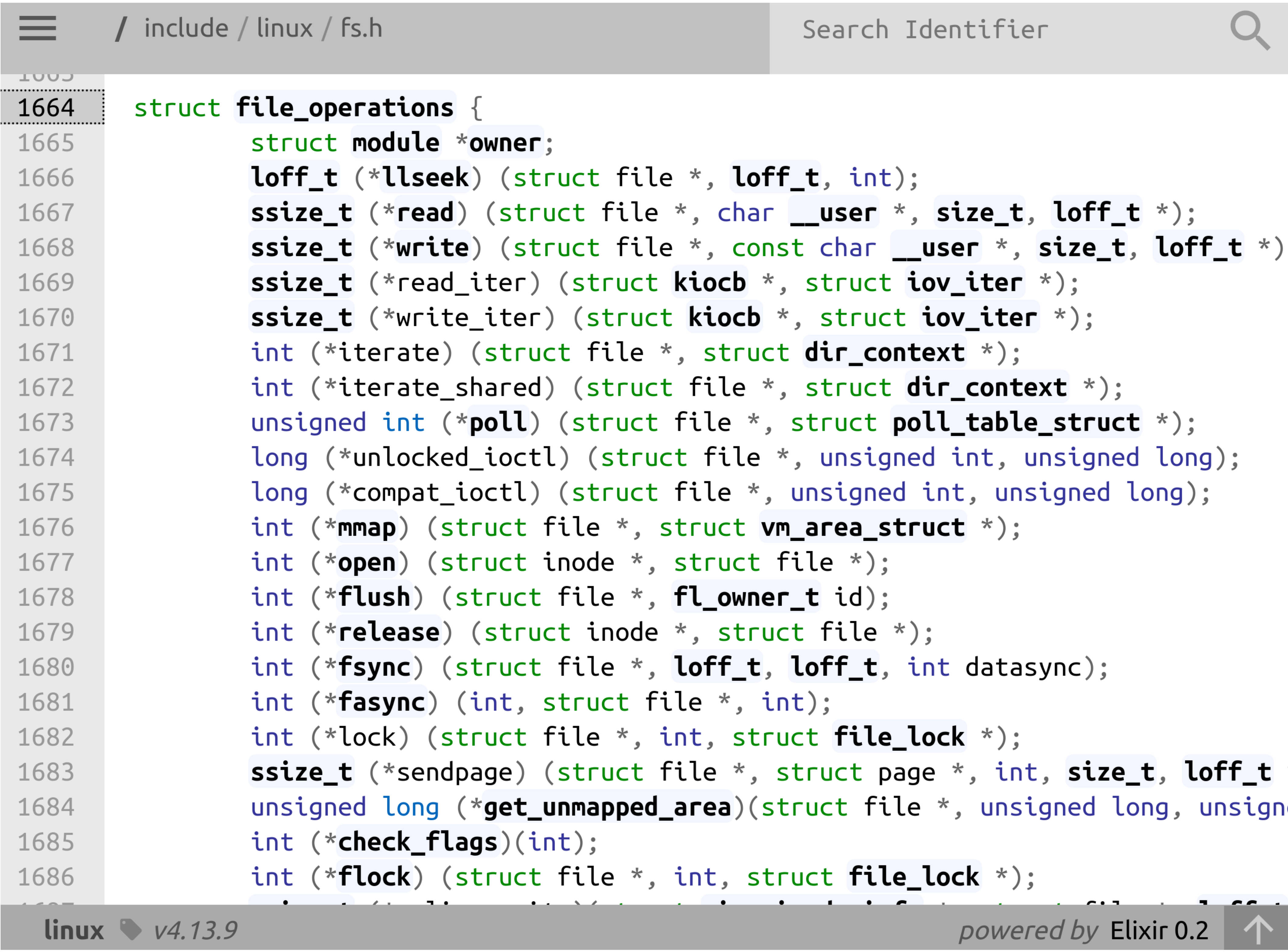


```
struct file {
        union {
                struct llist_node       fu_llist;
                struct rcu_head         fu_rcuhead;
        } f_u;
        struct path             f_path;
        struct inode            *f_inode;       /* cached value */
        const struct file_operations    *f_op;

        /*
         * Protects f_ep_links, f_flags.
         * Must not be taken from IRQ context.
         */
        spinlock_t              f_lock;
        enum rw_hint            f_write_hint;
        atomic_long_t           f_count;
        unsigned int            f_flags;
        fmode_t                 f_mode;
        struct mutex            f_pos_lock;
        loff_t                  f_pos;
        struct fown_struct      f_owner;
        const struct cred       *f_cred;
        struct file_ra_state    f_ra;
```

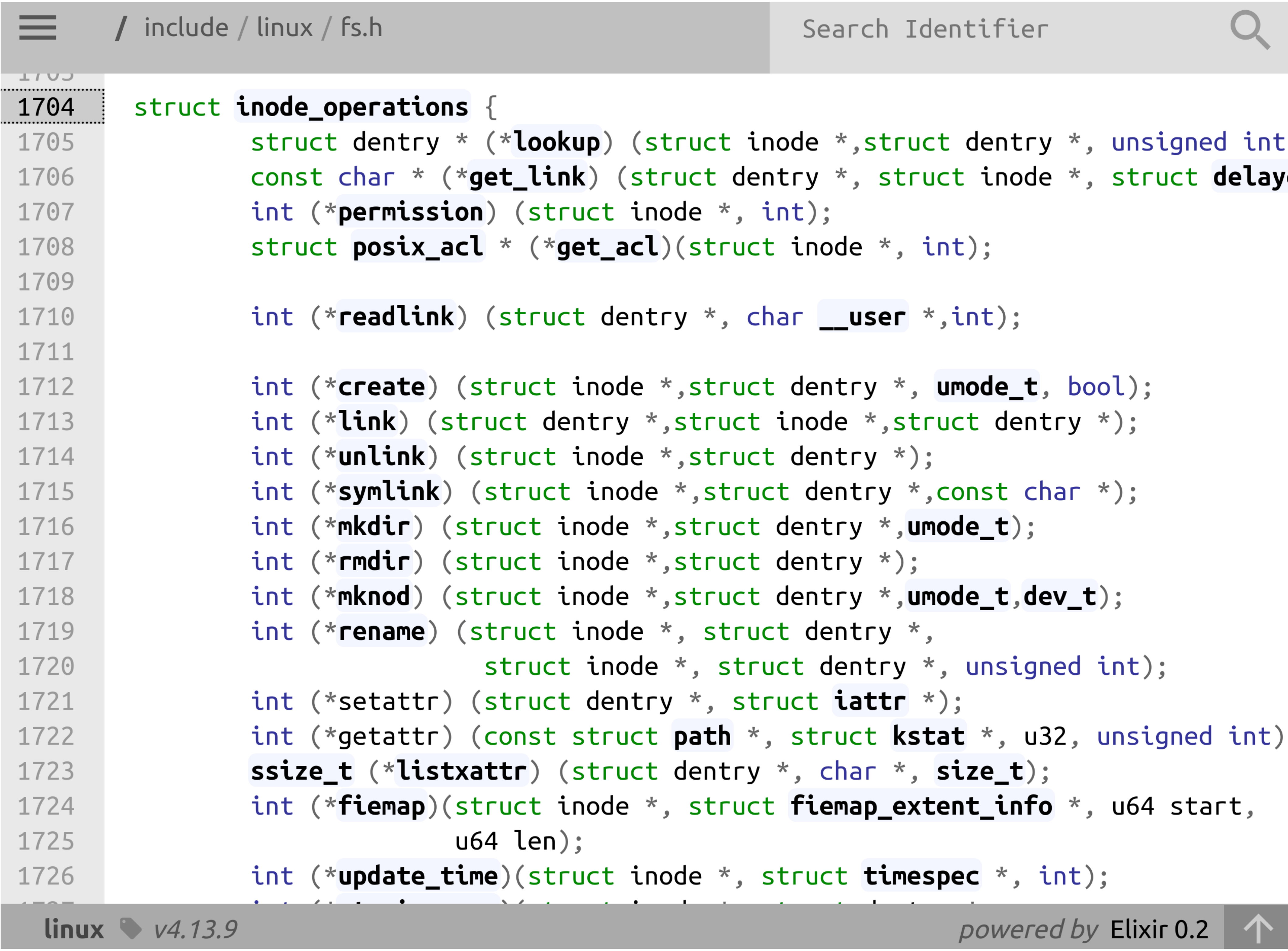


/ include / linux / fs.h

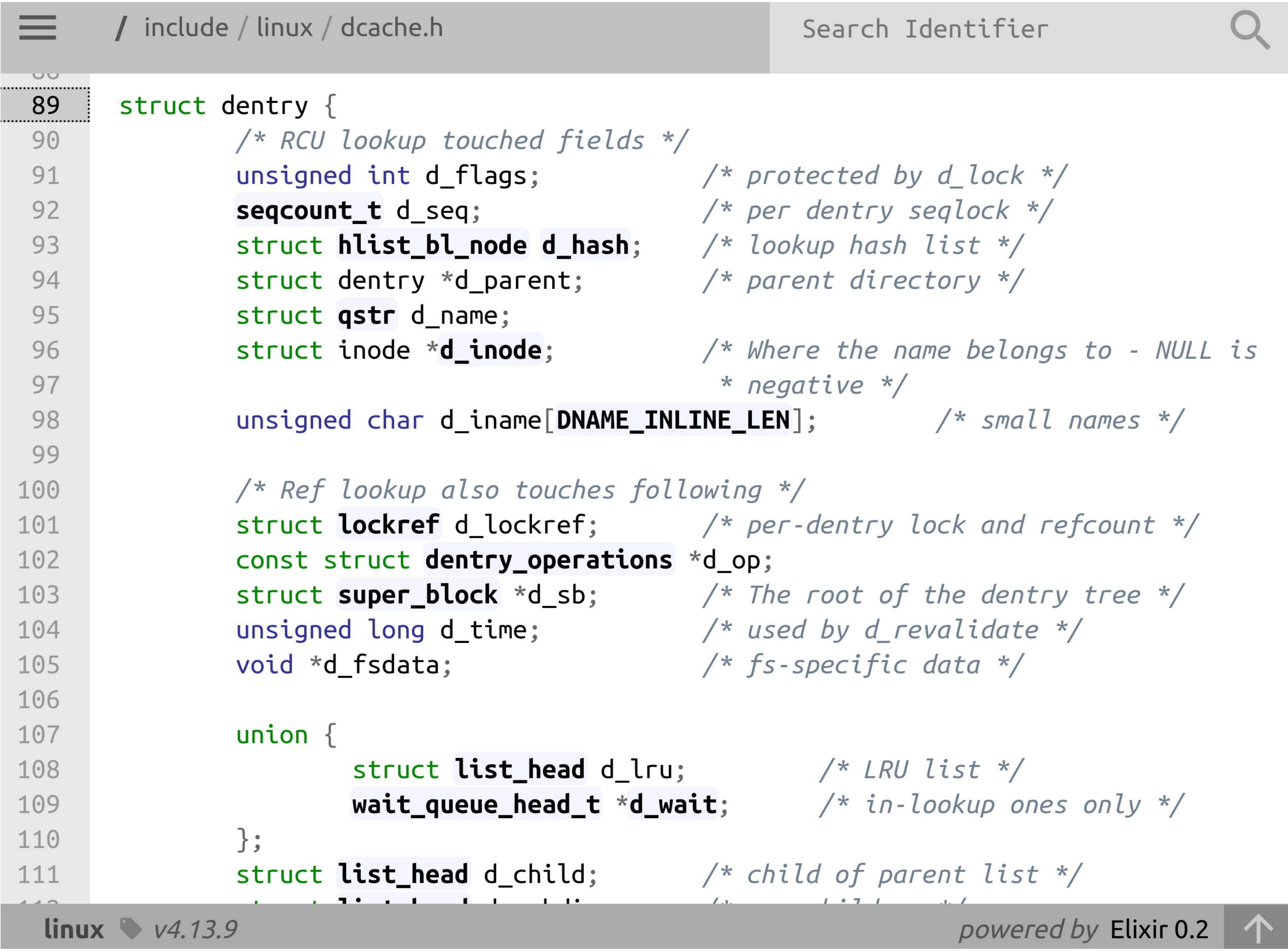
```c
/*
 * Keep mostly read-only and often accessed (especially for
 * the RCU path lookup and 'stat' data) fields at the beginning
 * of the 'struct inode'
 */
struct inode {
	umode_t			i_mode;
	unsigned short		i_opflags;
	kuid_t			i_uid;
	kgid_t			i_gid;
	unsigned int		i_flags;

#ifdef CONFIG_FS_POSIX_ACL
	struct posix_acl	*i_acl;
	struct posix_acl	*i_default_acl;
#endif

	const struct inode_operations	*i_op;
	struct super_block	*i_sb;
	struct address_space	*i_mapping;

#ifdef CONFIG_SECURITY
	void			*i_security;
#endif
	const struct file_operations	*i_fop;	/* former ->i_op->default_file_
```



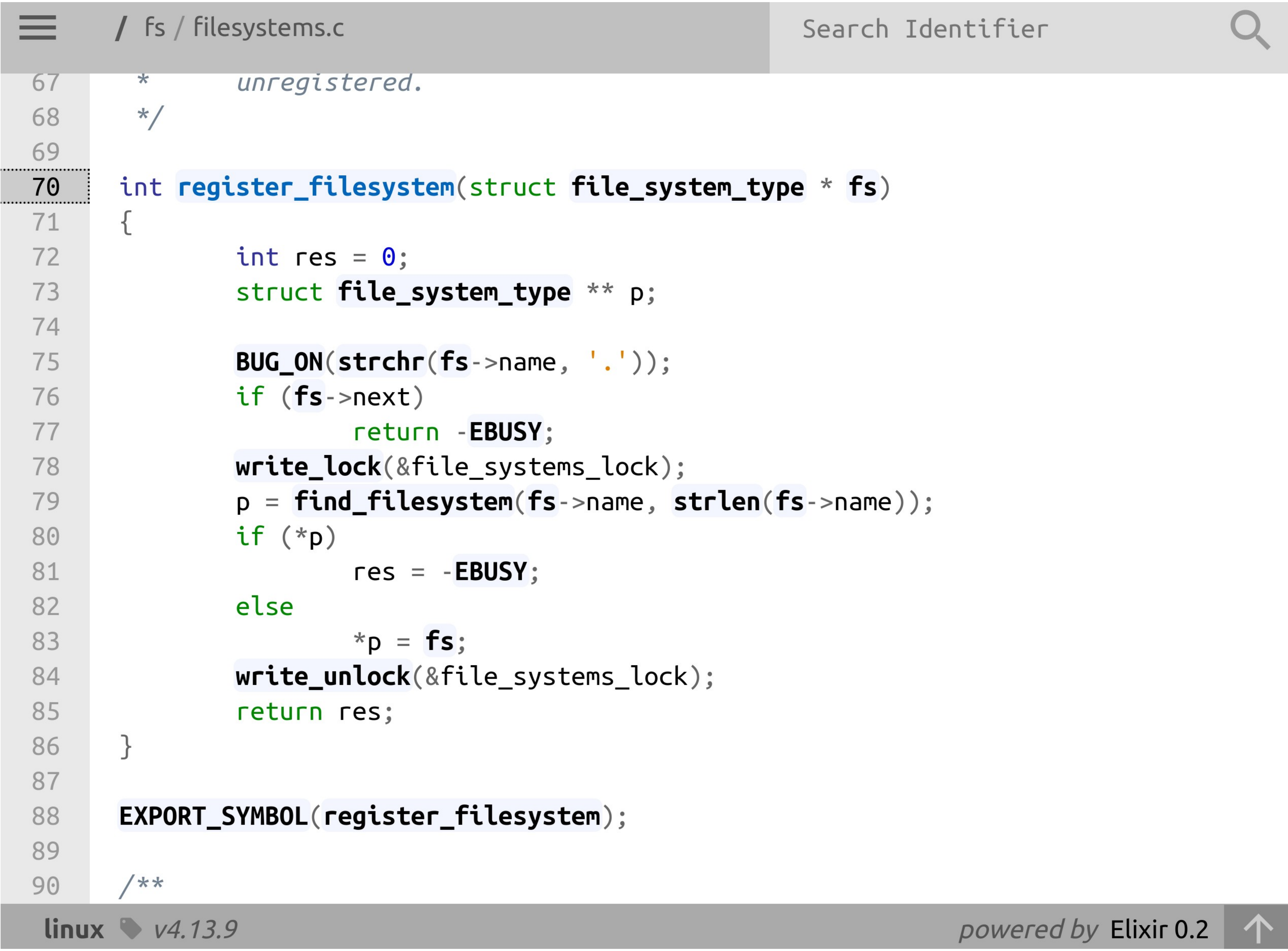
```c
struct file_operations {
        struct module *owner;
        loff_t (*llseek) (struct file *, loff_t, int);
        ssize_t (*read) (struct file *, char __user *, size_t, loff_t *);
        ssize_t (*write) (struct file *, const char __user *, size_t, loff_t *)
        ssize_t (*read_iter) (struct kiocb *, struct iov_iter *);
        ssize_t (*write_iter) (struct kiocb *, struct iov_iter *);
        int (*iterate) (struct file *, struct dir_context *);
        int (*iterate_shared) (struct file *, struct dir_context *);
        unsigned int (*poll) (struct file *, struct poll_table_struct *);
        long (*unlocked_ioctl) (struct file *, unsigned int, unsigned long);
        long (*compat_ioctl) (struct file *, unsigned int, unsigned long);
        int (*mmap) (struct file *, struct vm_area_struct *);
        int (*open) (struct inode *, struct file *);
        int (*flush) (struct file *, fl_owner_t id);
        int (*release) (struct inode *, struct file *);
        int (*fsync) (struct file *, loff_t, loff_t, int datasync);
        int (*fasync) (int, struct file *, int);
        int (*lock) (struct file *, int, struct file_lock *);
        ssize_t (*sendpage) (struct file *, struct page *, int, size_t, loff_t
        unsigned long (*get_unmapped_area)(struct file *, unsigned long, unsign
        int (*check_flags)(int);
        int (*flock) (struct file *, int, struct file_lock *);
```



```c
struct inode_operations {
	struct dentry * (*lookup) (struct inode *,struct dentry *, unsigned int
	const char * (*get_link) (struct dentry *, struct inode *, struct delay
	int (*permission) (struct inode *, int);
	struct posix_acl * (*get_acl)(struct inode *, int);

	int (*readlink) (struct dentry *, char __user *,int);

	int (*create) (struct inode *,struct dentry *, umode_t, bool);
	int (*link) (struct dentry *,struct inode *,struct dentry *);
	int (*unlink) (struct inode *,struct dentry *);
	int (*symlink) (struct inode *,struct dentry *,const char *);
	int (*mkdir) (struct inode *,struct dentry *,umode_t);
	int (*rmdir) (struct inode *,struct dentry *);
	int (*mknod) (struct inode *,struct dentry *,umode_t,dev_t);
	int (*rename) (struct inode *, struct dentry *,
			struct inode *, struct dentry *, unsigned int);
	int (*setattr) (struct dentry *, struct iattr *);
	int (*getattr) (const struct path *, struct kstat *, u32, unsigned int)
	ssize_t (*listxattr) (struct dentry *, char *, size_t);
	int (*fiemap)(struct inode *, struct fiemap_extent_info *, u64 start,
		      u64 len);
	int (*update_time)(struct inode *, struct timespec *, int);
```



```c
struct dentry {
        /* RCU lookup touched fields */
        unsigned int d_flags;           /* protected by d_lock */
        seqcount_t d_seq;               /* per dentry seqlock */
        struct hlist_bl_node d_hash;    /* lookup hash list */
        struct dentry *d_parent;        /* parent directory */
        struct qstr d_name;
        struct inode *d_inode;          /* Where the name belongs to - NULL is
                                         * negative */
        unsigned char d_iname[DNAME_INLINE_LEN];        /* small names */

        /* Ref lookup also touches following */
        struct lockref d_lockref;       /* per-dentry lock and refcount */
        const struct dentry_operations *d_op;
        struct super_block *d_sb;       /* The root of the dentry tree */
        unsigned long d_time;           /* used by d_revalidate */
        void *d_fsdata;                 /* fs-specific data */

        union {
                struct list_head d_lru;         /* LRU list */
                wait_queue_head_t *d_wait;      /* in-lookup ones only */
        };
        struct list_head d_child;       /* child of parent list */
```



/ fs / filesystems.c

Search Identifier

```c
 *      unregistered.
 */

int register_filesystem(struct file_system_type * fs)
{
	int res = 0;
	struct file_system_type ** p;

	BUG_ON(strchr(fs->name, '.'));
	if (fs->next)
		return -EBUSY;
	write_lock(&file_systems_lock);
	p = find_filesystem(fs->name, strlen(fs->name));
	if (*p)
		res = -EBUSY;
	else
		*p = fs;
	write_unlock(&file_systems_lock);
	return res;
}

EXPORT_SYMBOL(register_filesystem);

/**
```

linux v4.13.9 *powered by* Elixir 0.2