

libfakekey

0.3

Generated by Doxygen 1.13.2



---

<b>1 Topic Index</b>	<b>1</b>
1.1 Topics . . . . .	1
<b>2 File Index</b>	<b>3</b>
2.1 File List . . . . .	3
<b>3 Topic Documentation</b>	<b>5</b>
3.1 FakeKey - . . . . .	5
3.1.1 Detailed Description . . . . .	6
3.1.2 Function Documentation . . . . .	6
3.1.2.1 fakekey_init() . . . . .	6
3.1.2.2 fakekey_press() . . . . .	6
3.1.2.3 fakekey_press_keysym() . . . . .	6
3.1.2.4 fakekey_release() . . . . .	7
3.1.2.5 fakekey_reload_keysyms() . . . . .	7
3.1.2.6 fakekey_repeat() . . . . .	7
3.1.2.7 fakekey_send_keyevent() . . . . .	8
<b>4 File Documentation</b>	<b>9</b>
4.1 fakekey.h . . . . .	9
<b>Index</b>	<b>11</b>



# Chapter 1

## Topic Index

### 1.1 Topics

Here is a list of all topics with brief descriptions:

FakeKey - . . . . .	5
---------------------	---



## Chapter 2

# File Index

### 2.1 File List

Here is a list of all documented files with brief descriptions:

<a href="#">fakekey.h</a> . . . . .	9
-------------------------------------	---





# Chapter 3

## Topic Documentation

### 3.1 FakeKey -

yada yada yada

#### Typedefs

- typedef struct [FakeKey](#) **FakeKey**  
*Opaque structure used for all operations.*

#### Enumerations

- enum [FakeKeyModifier](#)  
*enumerated types for #mb\_pixbuf\_img\_transform*

#### Functions

- [FakeKey](#) \* [fakekey\\_init](#) (Display \*xdpy)  
*Initiates [FakeKey](#).*
- int [fakekey\\_press](#) ([FakeKey](#) \*fk, const unsigned char \*utf8\_char\_in, int len\_bytes, int modifiers)  
*Sends a Keypress to the server for the supplied UTF8 character.*
- void [fakekey\\_repeat](#) ([FakeKey](#) \*fk)  
*Repeats a press of the currently held key ( from [fakekey\\_press](#) )*
- void [fakekey\\_release](#) ([FakeKey](#) \*fk)  
*Releases the currently held key ( from [fakekey\\_press](#) )*
- int [fakekey\\_reload\\_keysyms](#) ([FakeKey](#) \*fk)  
*Resyncs the internal list of keysyms with the server.*
- int [fakekey\\_press\\_keysym](#) ([FakeKey](#) \*fk, KeySym keysym, int flags)  
*[fakekey\\_press](#) but with an X keysym rather than a UTF8 Char.*
- int [fakekey\\_send\\_keyevent](#) ([FakeKey](#) \*fk, KeyCode keycode, Bool is\_press, int modifiers)

### 3.1.1 Detailed Description

yada yada yada

Always remember to release held keys

### 3.1.2 Function Documentation

#### 3.1.2.1 fakekey\_init()

```
FakeKey * fakekey_init (
    Display * xdp)
```

Initiates [FakeKey](#).

##### Parameters

<i>xdp</i>	X Display connection.
------------	-----------------------

##### Returns

new [FakeKey](#) reference on success, NULL on fail.

#### 3.1.2.2 fakekey\_press()

```
int fakekey_press (
    FakeKey * fk,
    const unsigned char * utf8_char_in,
    int len_bytes,
    int modifiers)
```

Sends a Keypress to the server for the supplied UTF8 character.

##### Parameters

<i>fk</i>	<a href="#">FakeKey</a> refernce from <a href="#">fakekey_init</a>
<i>utf8_char_in</i>	Pointer to a single UTF8 Character data.
<i>len_bytes</i>	Lenth in bytes of character, or -1 in ends with 0
<i>modifiers</i>	OR'd list of <a href="#">FakeKeyModifier</a> modifiers keys to press with the key.

##### Returns

#### 3.1.2.3 fakekey\_press\_keysym()

```
int fakekey_press_keysym (
    FakeKey * fk,
    KeySym keysym,
    int flags)
```

[fakekey\\_press](#) but with an X keysym rather than a UTF8 Char.

## Parameters

<i>fk</i>	<a href="#">FakeKey</a> refernce from <a href="#">fakekey_init</a>
<i>keysym</i>	X Keysym to send
<i>flags</i>	

## Returns

**3.1.2.4 fakekey\_release()**

```
void fakekey_release (  
    FakeKey * fk)
```

Releases the currently held key ( from [fakekey\\_press](#) )

## Parameters

<i>fk</i>	<a href="#">FakeKey</a> refernce from <a href="#">fakekey_init</a>
-----------	--

**3.1.2.5 fakekey\_reload\_keysyms()**

```
int fakekey_reload_keysyms (  
    FakeKey * fk)
```

Resyncs the internal list of keysyms with the server.

Should be called if a MappingNotify event is recieved.

## Parameters

<i>fk</i>	<a href="#">FakeKey</a> refernce from <a href="#">fakekey_init</a>
-----------	--

## Returns

**3.1.2.6 fakekey\_repeat()**

```
void fakekey_repeat (  
    FakeKey * fk)
```

Repeats a press of the currently held key ( from [fakekey\\_press](#) )

**Parameters**

<i>fk</i>	<a href="#">FakeKey</a> refernce from <a href="#">fakekey_init</a>
-----------	--

**3.1.2.7 fakekey\_send\_keyevent()**

```
int fakekey_send_keyevent (  
    FakeKey * fk,  
    KeyCode keycode,  
    Bool is_press,  
    int modifiers)
```

**Parameters**

<i>fk</i>	<a href="#">FakeKey</a> refernce from <a href="#">fakekey_init</a>
<i>keycode</i>	X Keycode to send
<i>is_press</i>	Is this a press ( or release )
<i>modifiers</i>	

**Returns**

## Chapter 4

# File Documentation

### 4.1 fakekey.h

```
00001 #ifndef _HAVE_LIBFAKEKEY_H
00002 #define _HAVE_LIBFAKEKEY_H
00003
00004 #include <stdio.h>
00005 #include <stdlib.h>
00006 #include <X11/X.h>
00007 #include <X11/Xlib.h>
00008 #include <X11/Xlibint.h>
00009 #include <X11/Xutil.h>
00010 #include <X11/cursorfont.h>
00011 #include <X11/keysymdef.h>
00012 #include <X11/keysym.h>
00013 #include <X11/extensions/XTest.h>
00014 #include <X11/Xos.h>
00015 #include <X11/Xproto.h>
00016
00017 #ifdef __cplusplus
00018 extern "C" {
00019 #endif
00020
00028
00029
00035 typedef struct FakeKey FakeKey;
00036
00042 typedef enum
00043 {
00044     FAKEKEYMOD_SHIFT    = (1<<1),
00045     FAKEKEYMOD_CONTROL  = (1<<2),
00046     FAKEKEYMOD_ALT      = (1<<3),
00047     FAKEKEYMOD_META     = (1<<4)
00048 } FakeKeyModifier;
00050
00058 FakeKey*
00059 fakekey_init(Display *xdpy);
00060
00061
00074 int
00075 fakekey_press(FakeKey          *fk,
00076               const unsigned char *utf8_char_in,
00077               int                 len_bytes,
00078               int                 modifiers);
00079
00085 void
00086 fakekey_repeat(FakeKey *fk);
00087
00088
00094 void
00095 fakekey_release(FakeKey *fk);
00096
00105 int
00106 fakekey_reload_keysyms(FakeKey *fk);
00107
00117 int
00118 fakekey_press_keysym(FakeKey *fk,
00119                      KeySym   keysym,
00120                      int      flags);
```

```
00121
00131 int
00132 fakekey_send_keyevent(FakeKey *fk,
00133                         KeyCode  keycode,
00134                         Bool      is_press,
00135                         int       modifiers);
00136
00137
00138
00139 #ifdef __cplusplus
00140 }
00141 #endif
00142
00143 #endif /* _HAVE_LIBFAKEKEY_H */
```

# Index

- FakeKey -, [5](#)
  - fakekey\_init, [6](#)
  - fakekey\_press, [6](#)
  - fakekey\_press\_keysym, [6](#)
  - fakekey\_release, [7](#)
  - fakekey\_reload\_keysyms, [7](#)
  - fakekey\_repeat, [7](#)
  - fakekey\_send\_keyevent, [8](#)
- fakekey.h, [9](#)
- fakekey\_init
  - FakeKey -, [6](#)
- fakekey\_press
  - FakeKey -, [6](#)
- fakekey\_press\_keysym
  - FakeKey -, [6](#)
- fakekey\_release
  - FakeKey -, [7](#)
- fakekey\_reload\_keysyms
  - FakeKey -, [7](#)
- fakekey\_repeat
  - FakeKey -, [7](#)
- fakekey\_send\_keyevent
  - FakeKey -, [8](#)