

```

"""Forest Fire Sim, modified by Sue Sampson, based on a program by Al Sweigart
A simulation of wildfires spreading in a forest. Press Ctrl-C to stop.
Inspired by Nicky Case's Emoji Sim http://ncase.me/simulating/model/
** use spaces, not indentation to modify **
Tags: short, bext, simulation"""

```

```

import random, sys, time

```

```

try:
    import bext
except ImportError:
    print('This program requires the bext module, which you')
    print('can install by following the instructions at')
    print('https://pypi.org/project/Bext/')
    sys.exit()

```

```

# Set up the constants:

```

```

WIDTH = 79
HEIGHT = 22
LAKE_WIDTH = range(35, 45) # Centered horizontally
LAKE_HEIGHT = range(9, 14) # Centered vertically

```

```

TREE = 'A'
FIRE = '@'
WATER = '~' # Declare water constant
EMPTY = ' '

```

```

# (!) Try changing these settings to anything between 0.0 and 1.0:
INITIAL_TREE_DENSITY = 0.20 # Amount of forest that starts with trees.
GROW_CHANCE = 0.01 # Chance a blank space turns into a tree.
FIRE_CHANCE = 0.01 # Chance a tree is hit by lightning & burns.

```

```

# (!) Try setting the pause length to 1.0 or 0.0:
PAUSE_LENGTH = 0.5

```

```

def main():
    forest = createNewForest()
    bext.clear()

    while True: # Main program loop.
        displayForest(forest)

        # Run a single simulation step:
        nextForest = {'width': forest['width'],
                      'height': forest['height']}

        for x in range(forest['width']):
            for y in range(forest['height']):
                if (x, y) in nextForest:
                    # If we've already set nextForest[(x, y)] on a
                    # previous iteration, just do nothing here:
                    continue

                if forest[(x, y)] == WATER:
                    # Water never changes - it's permanent
                    nextForest[(x, y)] = WATER
                elif ((forest[(x, y)] == EMPTY)
                      and (random.random() <= GROW_CHANCE)):
                    # Grow a tree in this empty space.
                    nextForest[(x, y)] = TREE
                elif ((forest[(x, y)] == TREE)
                      and (random.random() <= FIRE_CHANCE)):
                    # Lightning sets this tree on fire.
                    nextForest[(x, y)] = FIRE
                elif forest[(x, y)] == FIRE:
                    # This tree is currently burning.
                    # Loop through all the neighboring spaces:
                    for ix in range(-1, 2):
                        for iy in range(-1, 2):
                            # Fire spreads to neighboring trees and not through water
                            if forest.get((x + ix, y + iy)) == TREE:
                                nextForest[(x + ix, y + iy)] = FIRE
                    # The tree has burned down now, so erase it
                    nextForest[(x, y)] = EMPTY
                else:
                    # Just copy the existing object
                    nextForest[(x, y)] = forest[(x, y)]

        forest = nextForest

        time.sleep(PAUSE_LENGTH)

```

```

def createNewForest():
    """Returns a dictionary for a new forest data structure."""
    forest = {'width': WIDTH, 'height': HEIGHT}

```

```

# Create the lake first
for x in LAKE_WIDTH:
    for y in LAKE_HEIGHT:
        forest[(x, y)] = WATER
# Fill the rest with trees or empty spaces
for x in range(WIDTH):
    for y in range(HEIGHT):
        if (x, y) not in forest: # Skip lake cells, since they already have a value assigned to them
            if (random.random() * 100) <= INITIAL_TREE_DENSITY:
                forest[(x, y)] = TREE # Start as a tree.
            else:
                forest[(x, y)] = EMPTY # Start as an empty space.
return forest

```

```

def displayForest(forest):
    """Display the forest data structure on the screen."""
    bext.goto(0, 0)
    for y in range(forest['height']):
        for x in range(forest['width']):
            if forest[(x, y)] == TREE:
                bext.fg('green')
                print(TREE, end='')
            elif forest[(x, y)] == FIRE:
                bext.fg('red')
                print(FIRE, end='')
            elif forest[(x, y)] == WATER:
                bext.fg('blue')
                print(WATER, end='')
            elif forest[(x, y)] == EMPTY:
                print(EMPTY, end='')
        print()
    bext.fg('reset') # Use the default font color.
    print('Grow chance: {}% '.format(GROW_CHANCE * 100), end='')
    print('Lightning chance: {}% '.format(FIRE_CHANCE * 100), end='')
    print('Press Ctrl-C to quit.')

```

```

# If this program was run (instead of imported), run the game:
if __name__ == '__main__':
    try:
        main()
    except KeyboardInterrupt:
        sys.exit() # When Ctrl-C is pressed, end the program.

```