WHEN BOOLEANS ARE NOT ENOUGH...
STATE MACHINES?

HARRINGTON JOSEPH

SR. SOFTWARE ENGINEER

TWITTER / GITHUB: **@harph**
SAN JOSE, CA

EXPECTATIONS

WHY, HOW AND WHEN TO USE STATE MACHINES

ANALOGY

- is_active
- active
- is_enabled
- enabled
- is_disabled
- disabled
- has_expired
- expired
- is_published
- published
- is_deleted
- deleted
- is_archived
- archived
- is_visible
- visible
- is_hidden
- hidden
- is_ready
- ready
- is_valid
- valid
- is_blocked
- blocked
- failed
- succeeded
- is_positive
- ...

THE PROBLEM

USING BOOLEANS TO REPRESENT STATES AND ENFORCE BUSINESS RULES

CODE

```
class Video:

    def __init__(self, source):
        self.source = source
        self.is_playing = False

    def pause(self):
        # Make the call to pause the video
        self.is_playing = False

    def play(self):
        # Make the call to play the video
        self.is_playing = True

    def stop(self):
        # Make the call to stop the video
        self.is_playing = False
```

```
video = Video('s3://video/storage/demo.mov')

video.play()

video.pause()

video.stop()

video.is_playing
```

```
video.is_playing
```

WHAT DOES IT REALLY MEAN?

```
video.is_playing
```

```
True
```

Video is playing

`video.is_playing`

False

Is it paused?

Is it stopped?

```
class Video:

    def __init__(self, source):
        self.source = source
        self.is_playing = False
        self.is_paused = False

    def pause(self):
        # Make the call to pause the video
        self.is_playing = False
        self.is_paused = True

    def play(self):
        # Make the call to play the video
        self.is_playing = True
        self.is_paused = False

    def stop(self):
        # Make the call to stop the video
        self.is_playing = False
        self.is_paused = False
```

Is the video playing?

```
video.is_playing
```

Is the video paused?

```
video.is_paused
```

Is the video stopped?

```
not video.is_playing and not video.is_paused
```

BUSINESS RULES

1. A video can only be played when is paused or stopped.
2. A video can only be paused when is playing.
3. A video can only be stopped when is playing or paused.

1. A video can only be played when is paused or stopped.

```
class Video:
    ...

    def play(self):
        if not self.is_playing or self.is_paused:
            # Make the call to play the video
            self.is_playing = True
            self.is_paused = False
        else:
            raise Exception(
                'Cannot play a video that is '
                'already playing.'
            )
```

2. A video can only be paused when is playing.

```
class Video:
    ...

    def pause(self):
        if self.is_playing:
            # Make the call to pause the video
            self.is_playing = False
            self.is_paused = True
        else:
            raise Exception(
                'Cannot pause a video that is '
                'not playing.'
            )
```

3. A video can only be stopped when is playing or paused.

```
class Video:
    ...

    def stop(self):
        if self.is_playing or self.is_paused:
            # Make the call to stop the video
            self.is_playing = False
            self.is_paused = False
        else:
            raise Exception(
                'Cannot stop a video that is '
                'not playing or paused.'
            )
```

THE CODE IS RAPIDLY BECOMING

- Complex.
- Bloated.
- Repetitive.
- Hard to test.

```
class Video:
    # States
    PLAYING = 'playing'
    PAUSED = 'paused'
    STOPPED = 'stopped'

    def __init__(self, source):
        self.source = source
        self.state = self.STOPPED
```

1. **A video can only be played when is paused or stopped.**

```
class Video:
    ...

    def play(self):
        if self.state != self.PLAYING:
            # Make the call to play the video
            self.state = self.PLAYING
        else:
            raise Exception(
                'Cannot play a video that is '
                'already playing.'
            )
```

2. A video can only be paused when is playing.

```
class Video:
    ...

    def pause(self):
        if self.state == self.PLAYING:
            # Make the call to pause the video
            self.state = self.PAUSED
        else:
            raise Exception(
                'Cannot pause a video that is '
                'not playing.'
            )
```

3. A video can only be stopped when is playing or paused.

```
class Video:
    ...

    def stop(self):
        if self.state != self.STOP:
            # Make the call to stop the video
            self.state = self.STOPPED
        else:
            raise Exception(
                'Cannot stop a video that is '
                'not playing or paused.'
            )
```

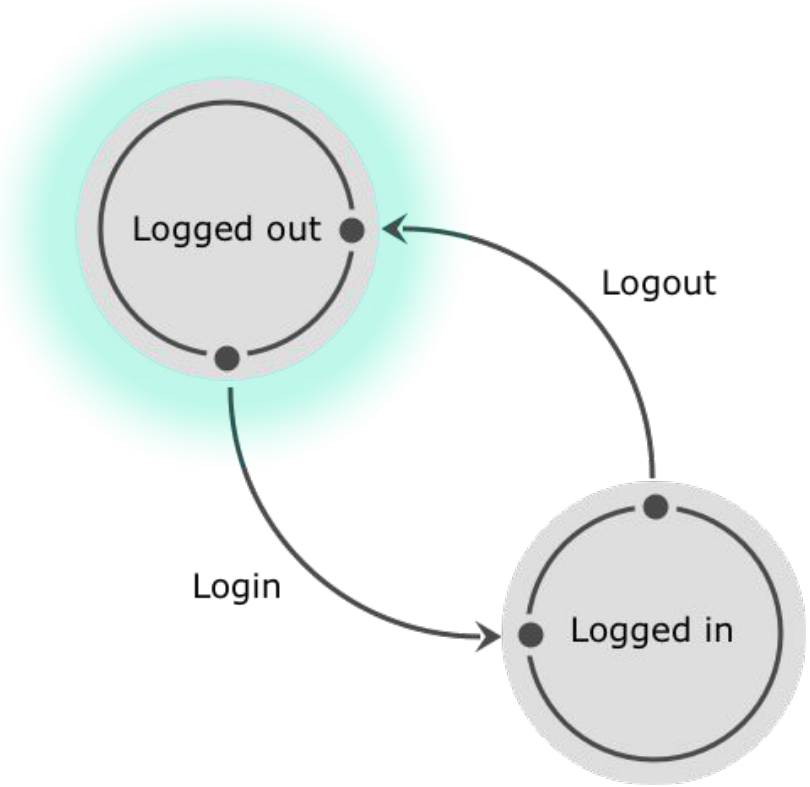
STATE MACHINES

WHAT'S A STATE MACHINE?

Mathematical model of computation.

- **Finite number of states.**
- **Transitions between states.**
- **Machine. Can only be in one state at a given time.**

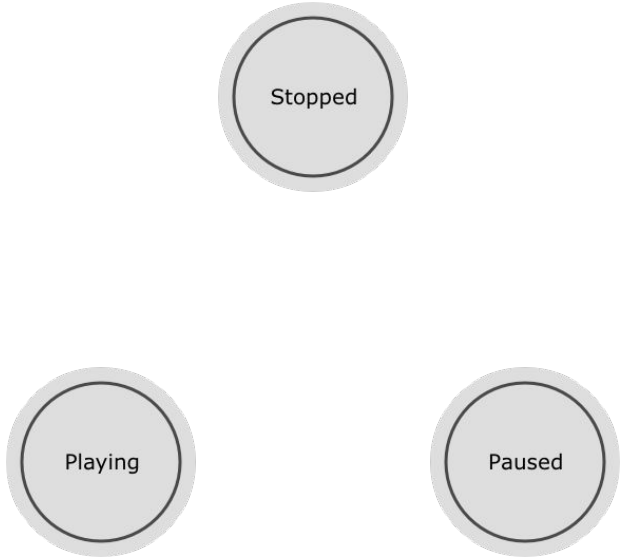
USER LOGIN STATE MACHINE



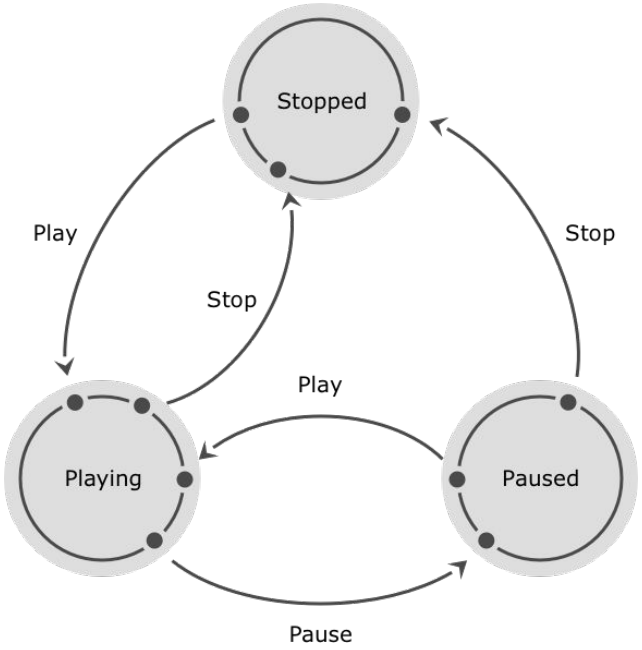
DESIGNING A STATE MACHINE

1. Define a finite number of states.
2. Lay down the transitions between states.
3. Select the initial state.

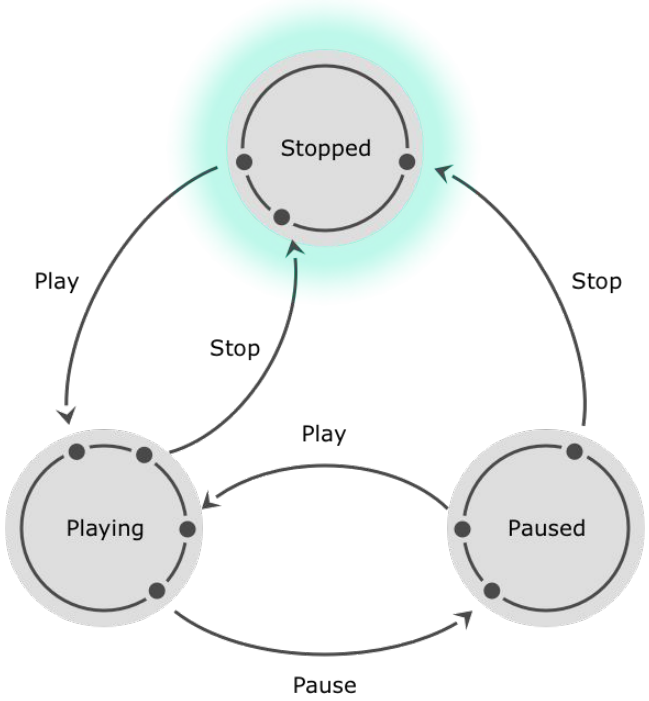
1. Define a finite number of states.



2. Lay down the transitions between states.



3. Select the initial state.



CODE

```
pip install transitions
```

github.com/pytransitions/transitions

```python
from transitions import Machine


class Video:
    # Define the states
    PLAYING = 'playing'
    PAUSED = 'paused'
    STOPPED = 'stopped'
```

```python
from transitions import Machine

class Video:
    # Define the states
    PLAYING = 'playing'
    PAUSED = 'paused'
    STOPPED = 'stopped'

    def __init__(self, source):
        self.source = source

        # Define the transitions
        transitions = [
            # 1. A video can only be played when is paused or stopped.
            {'trigger': 'play', 'source': self.PAUSED, 'dest': self.PLAYING},
            {'trigger': 'play', 'source': self.STOPPED, 'dest': self.PLAYING},
            # 2. A video can only be paused when is playing.
            {'trigger': 'pause', 'source': self.PLAYING, 'dest': self.PAUSED},
            # 3. A video can only be stopped when is playing or paused.
            {'trigger': 'stop', 'source': self.PLAYING, 'dest': self.STOPPED},
            {'trigger': 'stop', 'source': self.PAUSED, 'dest': self.STOPPED},
        ]
```

```python
from transitions import Machine

class Video:
    ...
    def __init__(self, source):
        self.source = source

        # Define the transitions
        transitions = [
            # 1. A video can only be played when is paused or stopped.
            {'trigger': 'play', 'source': self.PAUSED, 'dest': self.PLAYING},
            {'trigger': 'play', 'source': self.STOPPED, 'dest': self.PLAYING},
            # 2. A video can only be paused when is playing.
            {'trigger': 'pause', 'source': self.PLAYING, 'dest': self.PAUSED},
            # 3. A video can only be stopped when is playing or paused.
            {'trigger': 'stop', 'source': self.PLAYING, 'dest': self.STOPPED},
            {'trigger': 'stop', 'source': self.PAUSED, 'dest': self.STOPPED},
        ]
        # Create the state machine
        self.machine = Machine(
            model=self,
            transitions=transitions,
            initial=self.STOPPED
        )
```

```
class Video:
    ...

    def pause(self):
        # Make the call to pause the video
        ...

    def play(self):
        # Make the call to play the video
        ...

    def stop(self):
        # Make the call to stop the video
        ...
```

```
video = Video('s3://video/storage/demo.mov')

video.play()  # State is 'playing'

video.state  # 'playing'

video.pause()  # State is 'paused'

video.stop()  # State is 'stopped'

video.pause()  # ???
```

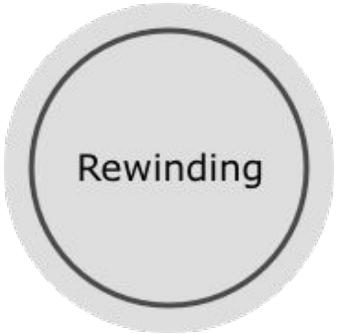
MachineError: "Can't trigger event pause from state stopped!"

TESTING

HOW DO WE TEST THIS?

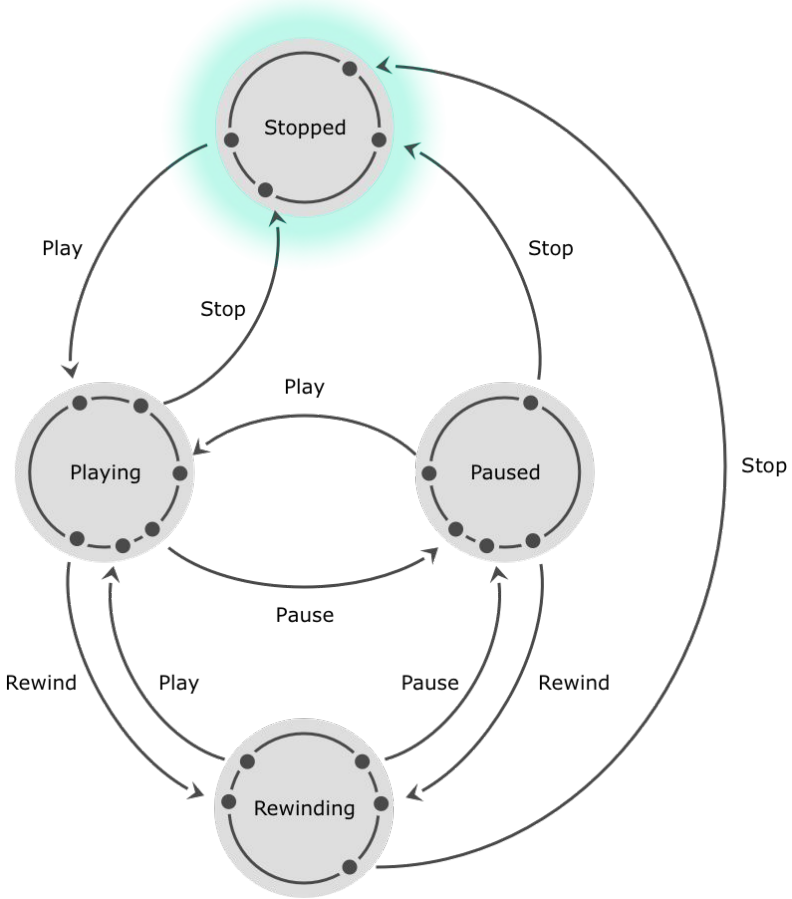
1. **We don't.**
2. Machine is initialized with the expected transitions and initial state.
3. Actual functionality (`play`, `pause` and `stop`).

ADDING A NEW STATE



RULES

1. A video can only be played when is paused, stopped or rewinding.
2. A video can only be paused when is playing or rewinding.
3. A video can only be stopped when is playing, paused or rewinding.
4. A video can only be rewinded when is playing or paused.



```python
from transitions import Machine

class Video:
    ...
    REWINDING = 'rewinding'

    def __init__(self, source):
        ...
        transitions = [
            # 1. A video can only be played when is paused, stopped or rewinding.
            {'trigger': 'play', 'source': self.PAUSED, 'dest': self.PLAYING},
            {'trigger': 'play', 'source': self.STOPPED, 'dest': self.PLAYING},
            {'trigger': 'play', 'source': self.REWINDING, 'dest': self.PLAYING},
            # 2. A video can only be paused when is playing or rewinding.
            {'trigger': 'pause', 'source': self.PLAYING, 'dest': self.PAUSED},
            {'trigger': 'pause', 'source': self.REWINDING, 'dest': self.PAUSED},
            # 3. A video can only be stopped when is playing, paused or rewinding.
            {'trigger': 'stop', 'source': self.PLAYING, 'dest': self.STOPPED},
            {'trigger': 'stop', 'source': self.PAUSED, 'dest': self.STOPPED},
            {'trigger': 'stop', 'source': self.REWINDING, 'dest': self.STOPPED},
            # 4. A video can only be rewinded when is playing or paused.
            {'trigger': 'rewind', 'source': self.PLAYING, 'dest': self.REWINDING},
            {'trigger': 'rewind', 'source': self.PAUSED, 'dest': self.REWINDING},
        ]
        ...
```

WHEN ARE BOOLEANS NOT ENOUGH?

- When multiple booleans represent a single state.
- When business rules are enforced by multiple booleans.

WHEN TO USE STATE MACHINES?

- When states are not binary.
- When you have to account for future states.
- When you have to enforce a complex set of business rules.

In summary, consider using state machines to represent states and enforce business rules

```
transitions = [
    # 1. A video can only be played when is paused, stopped or rewinding.
    {'trigger': 'play', 'source': self.PAUSED, 'dest': self.PLAYING},
    {'trigger': 'play', 'source': self.STOPPED, 'dest': self.PLAYING},
    {'trigger': 'play', 'source': self.REWINDING, 'dest': self.PLAYING},
    # 2. A video can only be paused when is playing or rewinding.
    {'trigger': 'pause', 'source': self.PLAYING, 'dest': self.PAUSED},
    {'trigger': 'pause', 'source': self.REWINDING, 'dest': self.PAUSED},
    ...
```

www.hjoseph.com
@harph