**Code Review Inputs:**

* Attempt to split up the code from one document
* What does Pygame.time do (explain in comment)
* Explain .unicode (comments likely)
* import files instead of long code
* get the pygame window to label itself as “trumpocalypse”
* reduce duplication in the text size menus
* Make health simpler and clearer for user
* Play with color scheme perhaps
* Book at store to describe vehicles, etc. maybe stat boost?
* Daily to do list, for events/requirements?

**Character.health -= \_\_\_**

Mauled at work…

* When leaving the WorkScreen, fire process\_events (See Chart), which checks Character.health.

When the month rolls over and all active events are processed…

When the month rolls over and daily\_things happens…

* Daily things?
  + StayingWithFriends
  + No food
  + No clothing
  + ...

When user first "activates" an event on EventScreen, it fires one time…

**Game sends alert when HP is 0, if no first aid kit then end game, if there is then use and regain health**

There is some coding setup for yes/no confirmation.

Maybe use this?

Yes: Good job

No: Oops, you died!

**-1 food per day, if no food -1 HP.**

**At the end of doing all of the mods… tell the user the character's current status… and at the same time display the FirstAidKits alert or DeathAlert if character is dead.**

=========================

game\_state.game.day.end\_mods()

=========================

def end\_mods(self):

# Mod character inventory items

game\_state.game.character.inventory.use\_daily\_items()

# Mod store items

game\_state.game.locations.restock\_stores()

# Anything else?

========================================

game\_state.game.character.inventory.use\_daily\_items()

========================================

def use\_daily\_items(self): # Mod character.

# House: Remove 30 days of use?

if game\_state.game.character.selected\_house == 'Staying with Friends':

hidx = game\_state.game.character.selected\_house\_idx

self.sorted\_items['housing'][hidx].remaining\_uses -= 30

# Lower sanity

...

# Food: Remove about 30 days \* 3 uses per day = 90

# Clothing: Remove 30 days \* 1 clothing = 30

================================

game\_state.game.locations.restock\_stores()

================================

def restock\_stores(self): # Mod stores.

for location in self.locations:

for store in location.stores:

store.inventory.restock\_daily\_items()

**(Not super important right now, but needs to be done at some point) Have to add in high score calculation for end game**

…

**Maybe add in overthrow govt option at election time (maybe)**

# Chances of successful overthrow?

# 10% base chance + \_\_\_\_\_\_\_\_\_\_\_\_\_ .

# +5% chance for each K=1 or I=1 or B=1 … up to +20%

# +1% chance for each HP > base HP … up to 5%…

# +1% chance for each Intelligence > base Intelligence … up to 5%…

# +1% chance for each Strength > base Strength … up to 5%…

# something about Trump's health / anger …

#

#

# … etc. ...

# Costs of overthrow?

# Win overthrow = points

# Lose overthrow = Lose the Game

**Add items to fight effects of events? Similar to first aid kit to fight death…? (maybe)**

How about instead of on CharacterHUD it is on events page?

Add a third option to "new" events page: Counteract Event. On EnterKey… If current event viewing is activated… then iterate through character.inventory.items. Look for all items where item counteracts event: items.n['counteract'][ item\_type ][ event.event\_text ] ... that is   
event.event\_text in items.n['counteract'][ item\_type ]. If it is in there then add as a button in an alert. The alert has all items counteracting this event. Then the user clicks on one. Chance happens. A result appears. And it says "Try again." or "That's good for now.".

If the user tries to counteract an event and there are no inventory items that counteract this event… then alert the user of this with an "OK" button to continue.

Then as the event is counteracted… the main text box on EventsScreen will be updating the event to show the latest event stats.

* *Note*: Consider... Grouped versus single. Grouped: "Guns: 64". Single: "Fighter Jet: 92%".

**Fight Effects**

Fight effects: Like guns for zombies? Sounds reasonable.

Maybe using four guns when zombie happens… each time it counteracts -1 of zombie detrimental effects. So zombie is: {"hours":-4,"sanity":-2,"income":-5000}. So use a gun to: chance(hours-1 OR sanity-1 OR income-1000). If hours, hours = -3. If sanity, sanity = -1. If income, income=-4000. Unless it is at zero. then it "fails" to work.

These levels stay until the event is over (months remaining = 0). So to completely counteract zombies… you need at least 4+2+5=11 guns. And then it is chance, so. Not like it is actually 11. That would be like a treatable disease: There is a way to completely get rid of it but it still has a few lingering side effects.

For Events then: each event needs to know items counteracting it.

(Assumption: Everything counteracts it evenly.)

And then for Items: each item needs to know Events counteracted.

Like… events\_counteracted = [ … ]

Like… items\_counteracting = [ … ]

The amount could be how many time it rolls the chance dice.

One dictionary to rule them all is ideal. (*Notice that there is an items.py file now*). So add to this.

...

'counteracts':

{

'Gun': { 'Some Event': 1, 'Another Event': 2 },

'Dead Fish': { 'Trump Tries to Kiss You': 3 },

}

…

Assumption: On gun.use: gun.amount -= 1.

Question: Would using some items require anything? Such as hours?

**Add werk ;) screen, that has a story about what happened during the day -- working on it**

~~For money…~~

~~game\_state.game.character.earn\_money( num\_hours )~~

~~def earn\_money ( self, num\_hours ):~~

~~amount = self.income \* (num\_hours / 8)~~

~~self.add\_item('Cash', amount)~~

#2 and #3, refactor jobs using classes:<https://docs.google.com/document/d/1DcmAoJAivDuh1i43Zne5URcIWtVoKRtGylFAAE7EgIc/edit>

**Make surface bigger to fit more events (maybe dynamically?) -- or just only allow 4 items to pile up**

<https://docs.google.com/document/d/1DcmAoJAivDuh1i43Zne5URcIWtVoKRtGylFAAE7EgIc/edit>

**When the stores re-stock (monthly), on the next visit there ought to be a picture of the shopkeep with a bubble saying "Well, Hello \_\_\_! Good to see you again. Here's a flyer of the latest (and greatest!) goods we just stocked up on. [maybe a small pygameui list]."**

...

**~~Events ought to combine if possible (e.g. curfew+curfew=buffed curfew).~~**

<https://docs.google.com/document/d/1DcmAoJAivDuh1i43Zne5URcIWtVoKRtGylFAAE7EgIc/edit>

**~~For Events, add bonuses\_by\_ratio dictionary, which multiplies the original amount by a ratio.~~**

'Cash':0.9 means set cash to 90% of present value.

'health:0.1 means set health to 10% of present value (bad!).

**Food in store inventories is going minus on new days??**

…-19 food -$19 ?