EventStorage

h{pt A class that is designed to manage and store events within the game.}c!pp

class EventStorage(name: str, location: str, *options: Option, fallback: Event = None, fallback_text: str = "There is nothing to do here")

The EventStorage class is designed to manage and store events within the game. It provides methods to handle event data, update event states, and manage event conditions.
The storage is able to select a single or a set of events based on their conditions and if they are fulfilled.
This Storage type stores the Event based on their select_type.

Parameters:

name: str

- The name of the storage.
- Used to reidentify the Storage

location: str

- The location the Storage should be part of.
- The location is either one of the games map locations or "misc"
- Possible values: "**bath**", "**beach**", "**cafeteria**", "**courtyard**", "**gym**", "**kiosk**", "**labs**", "**office_building**", "**school_building**", "**school_dormitory**", "**sports_field**", "**staff_lodges**", "**swimming_pool**", "**misc**"

*options: Option

- A set of options to configure the Storage
- A List of Options can be found here:

fallback: Event (default: None)

- A fallback Event that is called instead, in case none of the Events stored are available
- If fallback is None, a default fallback event is called using the locations background image and the provided fallback text

fallback_text: str (default: "There is nothing to do here.")

- The text that is displayed in case the Storage calls the default fallback event

Methods:

is_fulfilled(kwargs) -> bool**

Returns whether the stats of the character fulfill the condition

Parameters:

1. ****kwargs**
   - Additional arguments
   - Method possibly checks for key **'char_obj'** in **kwargs** looking for a Character Object

Returns:

1. **bool**
   - Whether the condition is fulfilled or not

to_list_text(kwargs) -> Tuple[str, str] | Tuple[str, str, str] | List[Tuple[str, str] | Tuple[str, str, str]]**

Returns the description text for the condition that is displayed in the display list.
If multiple stats are checked, the condition is displayed as a list.

Parameters:
1. ****kwargs**
   - Additional arguments
   - Method possibly checks for key **'char_obj'** in **kwargs** looking for a Character Object

Returns:

1. **Tuple[str, str] | Tuple[str, str, str] | List[Tuple[str, str] | Tuple[str, str, str]]**
   - The condition text for the display list.
   - The first element is the icon, the second element is the value and the third element is the title.
   - The title is optional.
   - Multiple conditions can be returned as a list.

to_desc_text(**kwargs) -> str | List[str]

Returns the description text for the condition that is displayed in the description.
If multiple stats are checked, the condition is displayed as a list.

Parameters:

1. ****kwargs**
   - Additional arguments
   - Method possibly checks for key '**char_obj**' in **kwargs** looking for a Character Object

Returns:

1. **str | List[str]**
   - The condition text for the description.
   - Multiple conditions can be returned as a list.

get_name() -> str

Returns the name of the condition.
If multiple stats are checked, the condition is displayed as a comma separated list.

Returns:

1. **str**
   - A List of checked Stats
   - Multiple stats are seperated by commas

get_diff(char_obj: Char) -> num

Returns the difference between the condition and the given character.
If the condition difference is lower than -20, the difference is multiplied by 10.
If the condition difference is lower than -10, the difference is multiplied by 5.

If the condition difference is lower than -5, the difference is multiplied by 2.
Otherwise the difference is returned as is.
This method is used to calculate the probability of the PTA approving.

Parameters:

1. **char_obj: Char**
   - The character to compare the condition to.

Returns:

1. **num**
   - The difference between the condition and the given character

Examples:

StatCondition("school", inhibition = 90)

StatCondition(True, corruption = "20-30")

StatCondition(happiness = "1,2,5,6")

StatCondition(education = "10+", charm = "90-")

StatCondition(reputation = "1,10-20,50+")