class VectorModel (Max)

func: __init__(self, vector_dict: Dict[str,np.ndarray])

func: embed(self, word: str) -> np.ndarray

func: vector_size(self) -> int

func: cosine_similarity(self, vec1: np.ndarray, vec2: np.ndarray) -> float

func: most_similar(self, word: str, top_n: int=5) -> List[Tuple[str, float]]

func: most_similar_vec(self, vec: np.ndarray, top_n: int=5) -> List[Tuple[str, float]]

class GameGenerator (Max)

func: __init__(self, possible_words: List[str])

func: generate_words(self, words_n: int = 25) -> List[str]

func: assign_belongings(self, words: List[str], bomb_n: int = 1, teamA_n: int = 9; teamB_n: int = 8) -> game_words: List[GameWord]

func: create_clue_giver_bot(self, game_words)

func: create_guesser_bot(self, game_words)

func: create_game_manager(self)

func: create_game_ui_creator(self)

func: start_game(self)

class GameUiCreator (Florian)

func: __init__(self, game_words: List[GameWord], player_is_guesser: bool)

func: create_game_ui(self)

class GameWord (Max)

func: __init__(self, word: str, belonging: enum)

func: store_most_similar_words(self, most_similar_words: List[str])

func: add_shared_similar_word(self, shared_similar_word: str, score: float, sharing_words: List[GameWord])

func: reveal_belonging(self)

class ClueWord (Florian)

func: __init__(self, word: str, scores: List[Tuple[GameWord,score])

func: sort_by_score(self)

func: get_clue_score(self,team) -> clue_score: Tuple[int,float]

func: set_clue_given(self)

class ClueGiverBot (Florian)

func: __init__(self,vector_model: VectorModel, team: enum, game_words: List[GameWord], similar_word_cutoff: int = 200)

func: get_most_similar_words(self, game_words)

func: get_shared_similar_words(self, game_words) -> possible_clues: List[ClueWord] (store internally)

func: get_best_clue(self) -> best_clue: ClueWord

func: give_clue(self)

class GuesserBot (Max)

func: __init__(self,vector_model: VectorModel, team: enum, game_words: List[GameWord])

func: take_guess(self, given_clue: Tuple(str,int))

func: handle_reveal(self, guess_was_right: bool)

func: store_wrong_guess(self, given_clue: Tuple(str,int))

func: take_extra_guess(self)

class WordButton (Florian)

func: __init__(self, bg_image: pygame_image, game_word: GameWord, row: int, col: int, player_is_guesser: bool, game_manager: GameManager)

func: draw(self)

func: clicked(self)

func: reveal_belonging(self)

class GameManager_PlayerIsGuesser (Florian)

func: __init__(self, game_words: List[GameWord], clue_giver_bot: ClueGiverBot)

func: start_game(self)

func: get_clue_from_bot(self)

func: handle_player_guess(self,pressed_button: WordButton)

func: check_for_game_end(self)

func: enable_next_guess(self)

func: end_turn(self)

func: end_game()

class GameManager_PlayerGivesClues (Max)

func: __init__(self, game_words: List[GameWord], guesser_bot: GuesserBot)

func: start_game(self)

func: ask_for_player_clue(self)

func: handle_player_clue(self,player_clue: Tuple(str,int))

func: get_guess_from_bot(self)

func: check_for_game_end(self)

func: end_turn(self)

func: end_game()