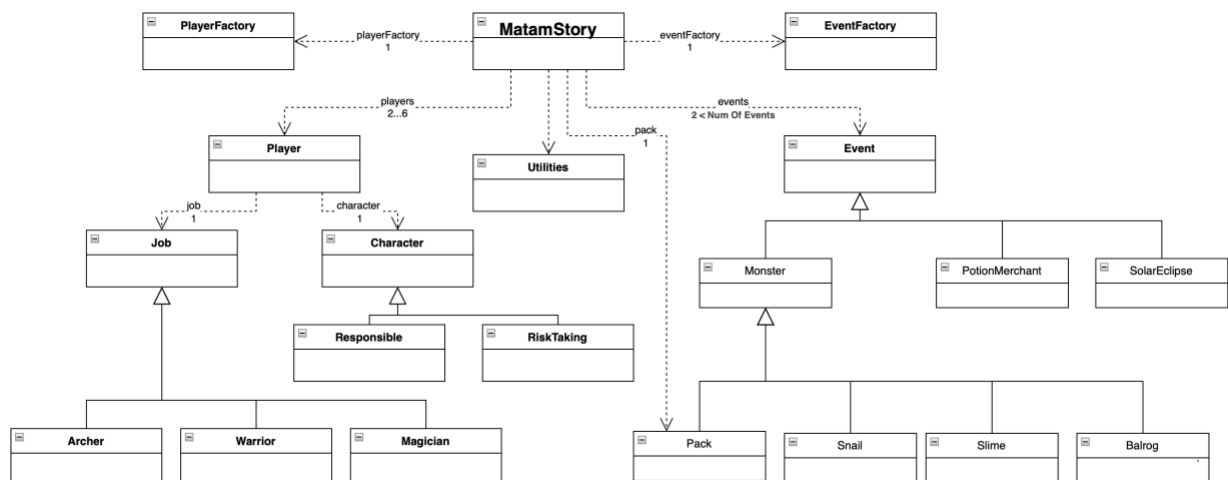




.1



תבנית Factory –

השחקנים והאירועים נוצרים בעזרת Factory.

תבנית Singleton –

המחלקה MatamStory היא מחלקה שאנו מעוניינים שתהיה רק אחת ממנה בכל פעם שמריצים משחק, כי היא אחראית לנהל אותו. לא נרצה שתהיה יותר ממחלקה אחת כי זה עלול לגרום התנהגות לא מוגדרת.

תבנית Composite –

אפשר לראות את התבנית הזאת בכך שיש לכל אובייקט מסוג שחקן שדות של Character ושל Job, הלוא הם אובייקטים בעצמם.

תבנית Strategy –

למשל המתודה calculateCombatPower אשר נועדה לחשב את ה-CP של כל דמות בהתחשב בסוג העבודה שלה, ובכללי השימוש של המחלקות Character ו-Job כמחלקות בסיס מופשטות.

תבנית Template –

המחלקה Monster יורשת מ-Event, כלומר היא עוקבת אחרי איזושהו טמפלייט כללי אך מרשה לטיפוסים מסוג Monster להתנהג בצורה מסוימת שמתאימה רק להם (על ידי דריסת מתודות).

3. ראשית, היינו מוסיפים מחלקה חדשה של Rouge אשר יורשת מ-Job (ומוסיפים את העבודה החדשה למפעל העבודות). לאחר מכן, הייתי מוסיף מתודה של Rouge שבודקת האם הדמות יכולה להתחמק מהקרב בהתאם להשוואה בין ה-CP שלה לשל המפלצת (מחזירה אמת או שקר). ב-applyBattle, המתודה שמבצעת קרבות, הייתי מוסיף בדיקה כאשר דמות אמורה להפסיד האם היא גנב בעזרת מתודת ה-getType שלה (כמו שיש בדיקה ל-Warrior אם דמות ניצחה כדי להוריד לו 10 נקודות חיים) ואם היא אכן מסוג גנב, אז הייתי מפעיל את

המתודה ממקודם ואם היא מחזירה אמת אז מסיים את הקרב, אם לא אז היא ממשיכה אל החלק בקוד שבו מפסידים רגיל.

אפרופו Warrior, עלינו להחליט האם Rouge נלחם מטווח קצר או רחוק כדי לדעת האם להוסיף אותו לאלו שיורדות להם נקודות חיים מלחימה בטווח קצר (פשוט להוסיף "או גנב" בבדיקה של "האם הוא לוחם").

4. ניתן לממש דרישה זו בקלות על ידי יצירת מופע חדש של העבודה החדשה (עטוף במצביע חכם, כמובן) והחלפת המופע של העבודה הקיימת בעבודה החדשה (מאחר שהעבודה אצלנו ממומשת כשדה של Player). נצטרך גם לטפל במקרים של החלפה לקשת או ללוחם, שכן נצטרך להוסיף 10 מטבעות או להוסיף 50 מקסימום נקודות חיים בהתאמה. תחת המתודה שתהיה אחראית על ההשראה השמימית, אפשר לבצע בדיקה זו וגם את ההתאמות הנדרשות במקרי הצורך.