שגב חן

322433400

התבניות עיצוב שבחרתי הן:

1. Observer
2. Decorator
3. Factory Method
4. Delegator

Observer .1

מדוע היא מתאימה למערכת?

במערכת שלי, יש אירועים מסוימים שדורשים עדכון של אובייקטים אחרים, כמו שינוי מצב החיה או התחרות שצריך לדווח עליו לאובייקטים אחרים. השופט במערכת שלנו יהיה מאזין, והחיות יהיו מואזנות.

מה היא מוסיפה למימוש?

התבנית מאפשרת למערכת שלי להיות דינאמית יותר, כאשר אובייקט מסוים משתנה, כל האובייקטים שתלויים בו מתעדכנים אוטומטית, זה תורם לגמישות קוד בכך שהוא מפחית את התלות בין האובייקטים ומאפשר להוסיף או להסיר observers מבלי לשנות את הקוד הקיים.

איך יש לממש את התבנית?

נממש את Observer במחלקות שאחראיות על תחרויות או חיות.

כאשר חיה משנה מצב (מתחילה לרוץ/סיימה את התחרות – לתחרות יחיד זה יהיה מיידי, לתחרות שליחים רק שהאחרון סיים), כל האובייקטים המתבוננים (observers) יתעדכנו בהתאם – האובייקטים המתבוננים יהיו: שופט.

נעדכן את מחלקת animalThread שתהיה Observable על ידי הוספת "extends Observable"

וכאשר החיה תסיים לרוץ אז היא תעדכן את הObserver שלה שהוא יהיה השופט על ידי notifyObservers.

נעדכן את מחלקת Referee שתהפוך לObserver ותוסיף את הניקוד של החיה לScoresMap שהתחרות מסתיימת.

נבצע שיוך של השופט כObserver של הanimalThread כדי שהשופט יקבל עדכון כאשר החיה מסיימת את התחרות.

2. Decorator

מדוע היא מתאימה למערכת?

במערכת שלי יש חיות ותחרויות, בעזרת תבנית העיצוב Decorator אוכל להוסיף פונקציונליות נוספת לחיות או לתחרויות, כמו חוקים נוספים בתחרויות או פרסים/מדליות, מבלי לשנות את הקוד הקיים שלהן.

מה היא מוסיפה למימוש?

Decorator תאפשר לי להוסיף פונקציונליות חדשה לאובייקטים קיימים בצורה דינאמית מבלי לשנות את הקוד הבסיסי שלהם, זה מאפשר להרחיב את המערכת בצורה קלה וללא התערבות בקוד הקיים.

איך יש לממש את התבנית?

אשתמש בDecorator על מנת שאוכל להוסיף מדליות לחיות בהתאם לתוצאה שלהן ולמיקום שלהן בתחרות.

נוסיף במחלקת Animal מתודה שתחזיר את מערך המדליות של החיה כך שאוכל להרחיב את המערך בעזרת Decorators.

ניצור מחלקת AnimalDecorator שירחיב את Animal על מנת שנוכל להוסיף תכונות לחיה בצורה דינאמית, נוסיף מתודה של getMedalArr על מנת שנקבל את המערך של המדליות של החיה.

ניצור מחלקת MedalDecorator להוספת מדליה לחיה שמרחיב/יורש מAnimalDecorator

מטרת המחלקה לבדוק את מיקום החיה לפי השדה של אותה החיה (score) ששומר את מיקום החיה/ הקבוצה בתחרות, ומוסיף מדליה בהתאם.

ההוספה מתבצעת על ידי בדיקה של התוצאה של החיה אם היא מקום ראשון נוסיף מדלית זהב למערך אם שני אז כסף ואם שלישי אז ארד, אחרת שום דבר.

3. Factory Method

מדוע היא מתאימה למערכת?

במערכת שלי יש סוגים שונים של תחרויות ובעלי חיים שצריכים להיווצר בהתאם לסוג התחרות וסוג החיה, הFactory Method יחליט בהתאם לקלט איזה חיה ליצור, אנו נרצה לבחור האם ליצור תנין כחיית יבשה או כחיית מים.

מה היא מוסיפה למימוש?

התבנית מאפשרת ליצור אובייקטים בצורה דינאמית ולנהל בקלות את התהליך של יצירת אובייקטים מורכבים מבלי לחשוף את הפרטים הפנימיים שלהם למשתמשים אחרים במערכת.

היא תורמת לגמישות בכך שהיא מאפשרת הוספת סוגים חדשים של בעלי חיים מבלי לשנות את הקוד הקיים.

איך יש לממש את התבנית?

ניצור ממשק לחיות בשם AnimalFactory שיכיל את המפעל ליצירת החיות השונות ויקבל את סוג התחרות.

המחלקה תיצור בהתאם לסוג התחרות Alligator עם ערך בוליאני true אם יבשה וערך false אם ימי.

4. Delegator

מדוע היא מתאימה למערכת?

במערכת שלי יש פונקציות מסוימות עם עיבוד נתונים וחישוב מדדים, נשתמש בDelegator על מנת לתת לסוגים שונים של חיות או תחרויות לטפל במשימות באופן שונה.

מה היא מוסיפה למימוש?

התבנית מאפשרת חלוקה ברורה של אחריות והפחתת מורכבות בקוד על ידי Delegate משימות לאובייקטים אחרים. זה משפר את תחזוקת הקוד ומאפשר גמישות ושינוי קוד קל יותר של התנהגות פונקציונלית מבלי לשנות את כל המערכת.

תבנית העיצוב Delegator מאפשרת להפריד את האחריות לפונקציות מסוימות ולממשן במחלקה נפרדת, במקום שמחלקת Alligator תצטרך לממש את כל הפונקציות בעצמה.

איך יש לממש את התבנית?

ניצור ממשק נוסף ייחודי לפונקציונליות חיית יבשה שAlligator יממש אותו.

ניצור מופע מסוג חיית יבשה עם כל הפרטים על Alligator ונוסיף את התכונה הנוספת לחיית יבשה שהיא מספר רגליים שיש לבעל החיים.

ובכך השתמשתי בDelegator כדי להעביר את האחריות על הפונקציונליות של חיית יבשה למחלקה נפרדת, זה מאפשר לי להפריד אחריות, לשמור על קוד נקי ומאורגן ולהפוך את הקוד לקל יותר להרחבה ותחזוקה.

**חלק ב':**

1. Singleton

סינגלטון היא תבנית עיצוב שמבטיחה שאובייקט אחד בלבד מסוג מסוים יתקיים במערכת ומספקת גישה גלובאלית אליו.

במערכת שלי יש אלמנטים דינמיים כמו חיות, שופטים, תחרויות, אין מצב שאובייקט בודד של מחלקה מסוימת צריך לשלוט או להיות הגלובלי היחיד.

ייתכן שיש מספר תחרויות במקביל או מספר חיות מסוג מסוים, ולכן שימוש בSingleton יהיה מגביל ולא הגיוני.

גם בהנתן הרחבות עתידיות, יהיה עדיין צורך בריבוי מופעים של אובייקטים שונים בו זמנית ולכן התבנית תפריע לגמישות ולא תתאים למבנה המערכת.

1. Mediator

התבנית היא תבנית עיצוב שמנהלת את התקשורת בין אובייקטים שונים ומרכזת את התקשורת שלהם דרך מתווך אחד בלבד, במורה להקטין את התלות בין האובייקטים השונים.

במערכת שלי, החיות מתחרות, שופטים מתבוננים, וניהול התחרויות נעשה בצורה דינאמית, כל אובייקט צריך יכולת תקשורת ישירה או אינטראקציה עם אובייקטים אחרים.

שימוש בתבנית ירכז את כל התקשורת במקום אחד, מה שעלול ליצור צוואר בקבוק בתקשורת בין אובייקטים ולהוסיף מורכבות מיותרת. זה גם יפחית את הגמישות של האובייקטים לפעול ולהגיב באופן עצמאי.

גם בהרחבות עתידיות סביר שיהיה צורך לשמור על תקשורת ישירה בין אובייקטים ולא להוסיף מתווך שמנהל את כל התקשורת, מה שיפגע ביכולת לנהל את האינטראקציות בצורה טבעית ומבוזרת.

1. Visitor

Visitor היא תבנית עיצוב שמאפשרת להוסיף פונקציות חדשות למחלקות מבלי לשנות את המחלקות עצמן. התבנית עושה זאת על ידי יצירת "מבקר" (Visitor) שמכיל את הפונקציונליות החדשה. המבקר "מבקר" באובייקטים השונים ומבצע בהם את הפעולה הנדרשת.

במערכת שלי, כל אובייקט (כגון חיה, תחרות או שופט) כבר מכיל פונקציונליות ייחודית משלו, והוספת פונקציות חדשות נעשית בדרך כלל דרך הרחבת המחלקות או הוספת מחלקות חדשות. השימוש ב-Visitor יכול להוסיף שכבת מורכבות נוספת, שבה צריך לתחזק מבקרים עבור כל פונקציה חדשה, מה שעלול לסרבל את המערכת. בנוסף, תבנית זו עלולה להפר את עקרון הכימוס בכך שהיא חושפת את המבנה הפנימי של המחלקות למבקרים החיצוניים.

גם בעתיד כאשר אזדקק להוסיף פונקציונליות חדשה, ארצה לעשות זאת בצורה ישירה בתוך המחלקות עצמן או באמצעות מחלקות יורשות, השימוש בVisitor יוסיף מורכבות מיותרת בניהול הפונקציות, במיוחד כאשר מדובר במערכת שבה האובייקטים הם בעלי פונקציונליות ספציפית וייחודית. Visitor מתאים למערכות שבהן המבנה של האובייקטים הוא קבוע, אך הפונקציונליות משתנה לעיתים קרובות, ההתנהגות הזו לא תואמת את המערכת שלי.