텍스트, 도표, 스크린샷, 라인이(가) 표시된 사진

자동 생성된 설명

* 게임 구조.
* TDD의 예시를 통해 확인할 사항들이 highlight 되어 있음.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

* 함수의 기능과 구조.
* 최소 단위의 테스트를 통해 단계별 구현이 이루어짐.

1. UserData 항목 관련 기초

|  |  |
| --- | --- |
| NewGame | 새로운 UserData를 만든다 |
| GetStatValue | UserData 내 특정 Stat 값을 반환한다 |
| 단계별 테스트 | 1. 새로운 UserData를 생성한다. 2. 해당 UserData에 초기값을 설정한다.   2-1. 해당 초기값들을 체크한다. |
| 테스트 단계 1 | * 데이터를 저장할 새로운 UserData 생성 |
| 테스트 단계 2 | * 정상적인 초기값(전체 StatValue 200 고정)으로 초기화되었는지 확인 * 결과 |

|  |  |
| --- | --- |
| ChangeStat | UserData내 특정 Stat값을 변화시킨다. |
| 단계별 테스트 | 1. 특정 Stat값을 일정 수준 증가시킨다. 2. 특정 Stat값을 일정 수준 감소시킨다. |
| 테스트 단계 1 | * phyStat을 50 증가시켜 증가 기능의 테스트 * 결과 |
| 테스트 단계 2 | * intStat을 50 감소시켜 감소 기능의 테스트 * 결과 |

2. StaminaCalc

|  |  |
| --- | --- |
| StaminaCalc | staStat 값에 따라 수치를 조절 |
| 단계별 테스트 | 1. staStat값이 일정 수치 이하일 때 전달받은 인자에 감소를 적용한다. 2. staStat값이 일정 수치 이상일 때 전달받은 인자에 증가를 적용한다. |
| 테스트 단계 1 | * 현재 Stat이 ReferenceStatValue의 절반일 때(Reference 값 미만인 경우)의 테스트. * 결과 * 본래 증가값인 8 의 절반으로 계산 |
| 테스트 단계 2 | * 현재 Stat이 ReferenceStatValue의 이상일 때 테스트. * 결과 |
| 테스트 단계 3(예외 사항 추가 테스트) | * 현재 Stat이 ReferenceStatValue와 같을 때 테스트. * 결과 |

1. StaminaCalcChange

|  |  |
| --- | --- |
| StaminaCalcChangeStat | Encapsulation을 위해 StaminaCalc의 직접적인 호출 없이 특정 Stat의 ID만 인자로 받아 해당 Stat을 계산된 증감폭을 반영하여 성장 |
| 단계별 테스트 | 1. 함수 호출 시 해당 stat이 성장한다. 2. Stat의 성장폭이 staminaCalc를 반영한 수치이다. |
| 테스트 단계 1 | * 함수가 호출되었을 때, 지정된 stat이 실제로 증가하였는지의 테스트 * 결과 |
| 테스트 단계 2 | * 기본 Stat으로 설정되었을 때, intStat을 인자로 전달하여 함수 호출 시 성장폭 반영의 테스트 * 결과 * 200인 staStat을 감안하여 기본 8의 증가값에서 감소된 6이 증가된 모습 |

텍스트, 스크린샷, 폰트, 라인이(가) 표시된 사진

자동 생성된 설명

* 함수의 기능과 구조.
* 최소 단위의 테스트를 통해 단계별 구현이 이루어짐.

1. 확률에 따른

|  |  |
| --- | --- |
| IsSuccessful | 확률을 전달받아 True False의 결과값을 반환 |
| 단계별 테스트 | 1. 0을 전달받았을 때 False를 반환한다. 2. 100을 전달받았을 때 True를 반환한다. 3. 50을 전달받았을 때 True False 반환 비율이 반영된다. |
| 테스트 단계 1 | * 0프로의 확률일 때 항상 False를 반환하는지의 테스트 * 결과 |
| 테스트 단계 2 | * 100프로의 확률일 때 항상 True를 반환하는지의 테스트 * 결과 |
| 테스트 단계 3 | * 50프로의 확률일 때 1000번의 iteration을 통하여 생성된 True False 비율이 50프로에 근사하는지의 테스트 * 해당 코드를 통해 확률 계산 * 결과 |

|  |  |
| --- | --- |
| PossibilityCalc | UserData내 특정 Stat의 값에 따라 확률을 계산하여 반환 |
| 단계별 테스트 | 1. Stat이 최대 일 때 최대 확률을 반환한다. 2. Stat이 최저 일 때 최저 확률을 반환한다. 3. 여러 Stat을 인자로 전달했을 때 해당 Stat을 기반으로 확률을 계산한다. 4. 기본 확률을 50%로 설정할 시 적용하여 계산한다. 5. Stat이 일정 수치 이하일 때 적용하여 확률을 반환한다. 6. Stat이 일정 수치 이상일 때 적용하여 확률을 반환한다. |
| 테스트 단계 1 | * 최대 Stat 값으로 설정했을 때 최대 확률 반환의 테스트 * 결과 |
| 테스트 단계 2 | * 최저 Stat 값으로 설정했을 때 최저 확률 반환의 테스트 * 결과 * Float value를 다뤄 사소한 차이 존재 |
| 테스트 단계 3 | * Stat 값이 기준 값 이상인 경우 기준 확률 이상 반환 여부의 테스트 * 결과 * Stat type을 speStat으로, startAtHalf를 true로 설정하여 여러 stat 인자의 전달, 기본 확률 50% 설정 시의 케이스도 테스트 |
| 테스트 단계 4 | * Stat 값이 기준 값 이하인 경우 기준 확률 이하 반환 여부의 테스트 * 결과 |
| 테스트 단계 5(예외 사항 추가 테스트) | * Stat이 기준 값과 동일한 경우 기준 확률 반환 여부의 테스트 * 결과 |

|  |  |
| --- | --- |
| GetKarma | 2번에 걸친 통과 테스트를 통해 Good, Normal, Bad의 결과를 반환 |
| 단계별 테스트 | 1. 설정된 Stat에 따라 예상과 같은 Good, Normal, Bad의 결과값을 반환한다. |
| 테스트 단계 1 | * 최대값으로 Stat 설정 시 2번에 통과 계산에 따른 확률로 각각 25%, 50%, 25%의 결과 반환 여부의 테스트 * 결과 |

장점:

* 분리된 단계별 Test로 개발 과정, 시스템 구조의 체계성 확립
* 테스트를 통해 의도된 기능의 정확한 구현 가능
* 블록 단위의 컴포넌트의 robust함 확보