applyBonus() Function

Written by Brian Immel on Jan 01, 2023

The `applyBonus` function allows you to apply in-game bonuses. This function can accept real-world weather data that can be used to apply weather effects as an additional bonus. The function can interface with iWeatherProvider which returns weather data. The `applyBonus` function can be used when an action has occurred during game play (e.g. attack, experience, etc.) and only accepts a float value (between 0 and 1).

If this is your first time using location awareness features, visit our location developer documentation and privacy notice to learn more about these features.

IWeatherProvider

`IWeatherProvider` returns a read-only enum describing the weather in the current location of the player that is approximately 5 square kilometers (well beyond the in-game viewport).

```
interface IWeatherProvider {
     WeatherStatus GetCurrentWeather(Location location);
}
```

`IWeatherProvider` returns the following enum types:

- Unset
- Clear
- Rainy
- Cloudy
- ErrorNotFound

The `applyBonus` function can be used with or without the data provided by `IWeatherProvider`.

applyBonus() with weather

If you want to apply an additional weather-based bonus, the `applyBonus` function can accept two parameters: `percentGood` (float: 0 - 1) and `weatherProvider` (object).

For example, if you choose to apply a “boosting” bonus based on clear weather in the player's current location, you could use the `applyBonus` function like this:

```
Void applyBonus (float percentGood, IWeatherProvider weatherProvider) {
     WeatherStatus status = weatherProvider.getCurrentWeather(getCurrentLocation());

     if (status == WeatherStatus.Clear) {
          percentGood += 0.2f;
     } else {
          percentGood -= 0.2f;
     }

     percentGood = Mathf.Clamp(percentGood, 0.0f, 1.0f);
     IBonus bonus = getBonus(percentGood);

     displayBonus(bonus);
}
```

The `applyBonus` function accepts a floating point number (0-1) and an object provided by `IWeatherProvider` that then returns the weather condition (see WeatherSatus options below for types and suggestions). If the weather status is clear, a value of 0.2 is added to `percentGood`. Any other status applies a value of -0.2. The `percentGood` value is then limited between the values of 0 and 1 as the weather and action values may modify the value below or above that range. From there, the bonus is calculated using the `getBonus` function and displayed to the player via the `displayBonus` function.

applyBonus() without weather

You can use the `applyBonus` function without location data (weather bonus) by submitting one parameter: `percentGood` (float).

```
void applyBonus(float percentGood) {
     IBonus bonus = getBonus(percentGood);

     displayBonus(bonus);
}
```

The `applyBonus` function applies to the `percentGood` value (via `getBonus` which has a valid range of 0 - 1) Then, this example displays a bonus (if any) to the player via the `displayBonus` function.

WeatherSatus options

Your `applyBonus` function should be able to handle the following types returned from the `WeatherStatus` enum. We provided some suggestions on how to apply it to your bonus calculation:

- Unset: `WeatherStatus` has no value (consider passing a value of 0)
- Clear: apply percentGood as you see fit
- Rainy: apply percentGood as you see fit
- Cloudy: apply percentGood as you see fit
- ErrorNotFound: `WeatherStatus` returned an error (consider passing a value of 0)

Error messages and help

While we have thoroughly tested this sample code, we cannot guarantee that it will work as expected. Environment settings and other modules may affect the outcome of this function. See our Error Messages guide for error message and troubleshooting assistance.

If our Error Messages guide doesn't provide a solution for your issues, review our online developer communities:

- Niantic Dev Portal
- StackOverflow
- Reddit

If you have a premier account, please log into our help center and file a request. We'll reply in 1-2 business days.