Player Storage

BaseIO provides an easy way to store data for the players. To use this feature just create a new class extending `com.unitedworldminers.BaseIO.storage.PlayerStorage`. This will be your storage class. Every player will have an instance of this class. You have to implement `void onNewUser()`, this method will be called everytime BaseIO has to create a new user. With `getUUID()` you can always retrieve the players UUID. `onNewUser` allows to throw a `NewInstanceException` when you don't want to create a storage for that user. But be warned, when retrieving userdata you will have to check for *null* storages.

> BaseIO saves your storage based on your modID! Changing the modID will result in data loss!
> The data itself on the other hand won't be deleted. You can try to modify the storage for that, but there will be no support for this.

To now save some values you just create your fields in your storage class. Every field in that class will be stored, if you don't want this you can give the `transistent` attribute to that field.

An example storage might be:

```
public class MyStorage extends PlayerStorage {
    boolean aBool;
    String aString;
    transient String bString;

    @Override
    protected void onNewUser() throws NewInstanceException {
        aString = "UUID: " + getUUID();
        bString = "This value won't be saved";
    }
}
```

To get a player's storage instance you just use one of the methods of the `PlayerIO` class.

- `get(String modID, UUID uuid)` is the fastest way to get your storage. *ModID* is the id you defined in your *@Mod* annotation. Use this method for cases running multiple times per second (e.g. tick events).

- `get(UUID uuid)` will retrieve your modID automatically, which is more expensive. Use this method only for user-input purposes (e.g. in commands).
- `get(String name)` will retrieve the player's UUID by the name and uses the method above next.
- `get(ICommandSender sender)` is a simplified method for general purpose, but only EntityPlayer is allowed (*FakePlayer* is a subclass of *EntityPlayer*).

Additionally you can retrieve all storages of a player by calling `ensuredGet(UUID uuid)`.
Additionally you can retrieve all your storages by calling `getAll(Class clazz)` with *clazz* as your storage class.

> PlayerIO will save your storage automatically.
> To manually save or reload the data, use PlayerIO.save() or PlayerIO.load().

An example working with your storage:

```
ICommandSender caller;
MyStorage storage = PlayerIO.get(caller);
if (storage.aBool) {
    storage.bString = "Permission granted.";
    PlayerIO.save();
}
```

Revision #4
Created 10 May 2017 08:12:36 by deregges
Updated 28 November 2017 15:03:10 by deregges