THE ISLE OF ANSUR
BETWEEN SHADOWS & LIGHT

Official modding guide

Legend

D — directory
F — file

Types of mods

Isle of Ansur currently has four types of mods (or "packs" which is more correct name):
- statpack
- themepack
- worldpack
- scripts

Every mod can constitute one or more of those pack types, as they all interact with different part of the game. Let me shortly introduce them.

I. Statpack
Statpack is type of pack that builds the backbone of whole game by implementing data related to statistics - a core feature of RPG game!
Statpacks let you add new races, classes, religions, origins, attributes, skills, but will also allow for adding new creatures in the future.

II. Themepack
Themepack is usually a pack structured around game's look - it introduces colour theming, images seen as a background for menus, as well as GUI elements.

III. Worldpack
Worldpack is type of pack that is not yet available in the game, however in future updates it will be core part of worldbuilding of Isle of Ansur.
It will allow you to create locations, NPCs, quests, and basically all the environment game puts you in.

IV. Scripts
Scripts aren't pack type per se, because they usually serve as addition of features to one of previous pack types.
Additionally, their implementation right now is far from good or optimised, thus using them risks high possibility of resulting with buggy behaviour.

How to add mods to the game?

Adding mods to the game in *pre-alpha 3* requires manual task of building whole mod structure in game files - initially this version was meant to come with *pre-alpha 2* unpacker, but it was delayed due to heavy rewrite third version got.


In the future, it will be enough to pack your whole mod structure into .zip file and then put into *mods* directory - mods will be unpacked automatically during start of the game, as it was in second IoA release.

More on how to build the mod will be explained in next section.

Creating statpack

Statpacks are made in *stats* directory in your IoA folder. Their structure is similar to other packs:

⚙ Main IoA folder^D
    ⚙ stats^D
        ⚙ statpack ID^D
            ✳ contents
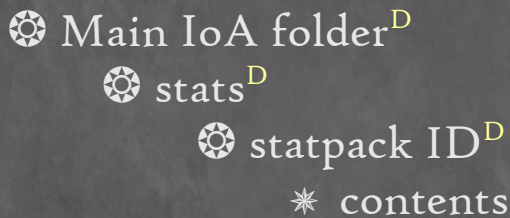
The ID is very important: it is what let the game distinguish between mods.
One mod can use multiple IDs, however this is rarely useful. What's more important is that IDs should be unique enough to not be overwritten by another mod.
IDs are currently unchangeable, by which it means that changing the ID will break saved games.

How to write proper ID?
Ideally, IDs should contain only lowercase latin letters and underscores instead of spaces - this ensures it is compatibile everywhere and does not produce buggy behaviour.
However, since there's no guardrails for it yet, game won't put any penalty on not following those rules, except for when you use ":" which is restricted keyword (it can't be put as folder name on Windows, so it shouldn't be even possible to use it for ID).

Statpack structure

While we discussed place where statpack is located, we haven't dived into how statpack structure looks like.

⚙ Statpack ID^D
    ⚙ lang^D
        ✳ english.toml^F
    ⚙ classes^D
    ⚙ races^D
    ⚙ origins^D
    ⚙ religions^D
    ✳ attributes.yaml^F
    ✳ skills.yaml^F
    ✳ perks.yaml^F
    ✳ info.toml^F

Two main parts of statpack is language file - located in *lang* folder - and statpack information file - being *info.toml* file. All other elements are entirely optional.

We will cover first one, since for now *info.toml* isn't used. This will change in next version.

Language file

Language files are TOML files that allow you to write translations of your mod elements to English (required) and other languages. They use the same naming as official languages used by IoA, meaning that there are only those languages available at the moment:

- English - being default and required option
- Polish - being indev language, not fully translated even in vanilla

Language file is structured around *keys*, which are semi-IDs made by you in mod features that guide them into specific text in language file.

Let me showcase an example.

```
custom_key = "Translated text"
    (A↑)              (B ↑)


    (A↓)              (B ↓)
another_key = '''
Multiline text, which can use
More lines than one and be formatted <b>with HTML</b>
(making newline uses <br> automatically, though!)
'''
```

In this example, we can see two elements: key (A) and content (B).
Just remember about this system, because it will be clearer how this
works once we create new element for IoA.

Speaking of which...

Creating Features

Statpack elements are made with use of two systems: JSON files
put into directories, and YAML files put in main directory.

I. Race

Creating race is made using JSON files. You do it simply by
creating *races* directory, and file of your choice with .json extension.

⚙ Statpack ID[D]
    ⚙ lang[D]
        ✳ english.toml[F]
    ⚙ races[D]
        ✳ example_race.json[F]
    ✳ info.toml[F]

Simply adding the file isn't enough and may result in even crashes, but that's how you initialise new races in general.
Remember that race filename is used to create unique race ID, consisting of [statpackID]:[race filename].

Once we created our JSON file, we should fill it with contents. The only thing you need to include as requirement is *key* field, but there are many fields you may want to use as well.

Field	Usage
key	Translation ID that points to respective key in language file
attributes	Attributes that this race has bonus for during character creation
power	Modifiers to four powers IoA bases some of statistics
skills	Skills this race has bonus for during character creation
magic	List of magical skill bonuses given to specific magic types during character creation
names	Lore-friendly names that will be suggested during character creation, separated by genders
perks	Perks that your race may use as their special ability

Feel free to refer to this cheatsheet above to implement fields and understand their purpose.

Below is example race file to understand how fields should look like and how you should write your JSON file.

```json
{
    "key": "example_race_key", (IA)
    "attributes": {                    (IB)
        "strength": 2,
        "agility":   2
    },
    "power": {                         (IC)
        "void": 2
    },
    "skills": {                        (ID)
        "archery":          2,
        "light_firearms": 1
    },
    "magic": {                         (IE)
        "kind": {
            "destruction": 2
        }
    },
    "names": {                         (IF)
        "male": [
            "Smith",
            "Gregory"
        ],
        "female": [
            "Anna",
            "Eleonora"
        ]
    },
    "perks": {                         (IG)
        "diamond_soul": 100
    }
}
```

IA. Translation key
While translation key system is pretty straightforward - you put your key (in our case, *example_race_key*) in race file, and then head to translation file and writes its translation:

```
example_race_key = "My Best Race Ever Created"
```

There's one small caveat.
You need also one more key being used in translation file, which is made out of your translation key and *_descr* being added.
This is description key and it serves purpose of describing your race to the player.
While it can be one-lined, it is common to use multilined string for it, so it is easier to format the description without using HTML <br> symbol too much.

Example translation file for our example can look like that then:

```
example_race_key        = "My Best Race Ever Created"
example_race_key_descr = '''
This race is very good at everything, but not perfect in anything!
It was created as God's creation like Frankenstein, made out of
pieces taken away from best mortals: Da Vinci, Einstein, Picasso
and thousands others!
Unfortunately somehow God made a mistake and while watching
Loki series, he put some trickstery to that race.
'''
```

IB. Attributes
Attributes are those big parts of player's character. They are used as a basis for many actions in game, and also decide heavily on skill improvements.
They can't be raised after character creation except for when your character levels up - so each attribute point pushes character in certain direction.

Attributes are made as a dictionary - those knowing code will already knowing what it means, but for those new to this concept, I'd say that this is exactly the system we are using already - so that pairing of *key* and *value*, like in *"key": "example_race_key"*.

There are only six attributes in vanilla IoA, but more can be added by mods. Balanced race build needs to sum attributes to 4 points (you can use negative values to balance).

Here is cheatsheet of attributes available in vanilla:

Attribute	Description	ID
strength	Strength affects melee fight effectivity, inventory load and other factors relying on muscle power.	ansur:strength
agility	Agility is about speed of various actions, so it tells how quick your character is performing actions.	ansur:agility
endurance	Endurance is art of resilience in situation where character has some physical tension affecting them.	ansur:endurance
charisma	Charisma is all about communication and personal impression on others.	ansur:charisma
perception	Perception tells how good character is in realising certain elements of environment around them.	ansur:perception
intelligence	Intelligence is huge factor on how easily character learns and how good at magic they are.	ansur:intelligence

What's important to mention is that you can use both name of attribute, as well as attribute ID for vanilla attributes - meaning that *strength* is equal to *ansur:strength*.
This is not the case with modded attributes, where ID needs to be explicitly written.

IC. Powers
Powers are currently very in-development aspect of character creation, but eventually they will have heavier impact on player interactions.

Baedoor universe operates on four forces, being opposites within their pairs:
MGCK is opposite to TECH, since magic destabilises technology in many cases, and VOID is opposite to CONN, as void energy is usually pure negation of balance, being chaotic creative force.

Powers set by races can take ranges from -20 to 20, and should not

have positive values when dealing with oppositions, unless perks allowing for it are also in place.

ID. Skills

Skills are minor part of races character development, as main skill aspect is dependant on classes. It is, however, often used, as races also can give additional bonus to skill points.

Balanced build is argued to have up to three skill points being distributed.

There are huge amount of skills added by vanilla, and similarly to attributes, they share similar behaviour of being ID-agnostic, if referred to core ones. Modded ones can be used as well, but they need explicit IDs.

Here is small cheatsheet of skills. For more detailed descriptions, please refer to the ones visible in-game.

Skill	Brief description
Combat category	
Handfight	Ability to fight without weapons
Shortswords	Ability to fight with light-weight bladed weapons
Longswords	Ability to fight with heavier bladed weapons
Axes	Ability to use axes and halberds in fight
Blunts	Ability to use maces and all other blunt weapons
Polearms	Ability to fight with spears
Aim	Ability to target precisely with ranged weapons
Throwing	Ability to use throwable weapons, like knives or javelins
Archery	Ability to use bows of various kinds
Crossbows	Ability to hit enemies with crossbows
Light Firearms	Ability to use revolvers and pistols
Heavy Firearms	Ability to fight with shotguns and rifles
Light Armour	Ability to wear leather armour
Medium Armour	Ability to use chainmails and other lighter metal armour
Heavy Armour	Ability to use heavier metal armours
Blocking	Ability to block damage with shield or weapon

Magic category	
Spellcasting	Ability to cast spells in general
Scrolls writing	Ability to bind spells into scrolls
Mores casting	Ability to bind rune spells into mores
Unsealing	Ability to unbind spells from items
Social category	
Trade	Your proficiency in trading, bargaining, recognising value
Persuasion	How well you can convince others of your views
Craft category	
Hunting	Skill on how well you can hunt and skin hunted animals
Fishing	Ability to find better places to fish and get more fishes
Sewing	Ability to make better clothes
Carpentry	Ability to craft items from wood
Pottery	Ability to create items from clay
Stonemasonry	Ability to make things from stone
Ore Processing	Ability to extract metal from ores and melt alloys
Smithing	Skill allowing you to create various items from metals
Herbalism	Allows you to recognise plants and their use
Alchemy	Ability to merge ingredients to craft useful potions
Cooking	Ability to cook ingredients into tasty meals
Personal category	
Wound Treatment	Ability to heal faster and protect yourself from harms
Horse Riding	Ability to ride a horse
Boat Driving	Ability to drive boats
Animal Taming	Ability to communicate with animals
Musicality	Ability to perform music play with various instruments
Athletics	Affects how well you perform on your body condition
Toughness	Affects your physical resistance
Survival	Ability to use extreme environments to your benefit

IDs of those skills are made by using lowercase for all characters and replacing spaces with _ character.

IE. Magic
Magic is subset of "invisible" magic skills being *common magic* and *path magic*. It is separated from skills because while being entirely beneficial (so you may think of lowering skill advantages if you add magic section to your race), it has very branched form.
In general, it is considered a good practice to either dedicate magic section entirely to common magic or path magic exclusively, however there isn't anything stopping you from doing both. Balanced race build sums up to six points on all magic branches. For further explanation, read further.

☀ Common Magic
This is kind of magic that is open for everyone and does not put any limitations on how points distribution affect your gameplay.

Common Magic works on combinations of element-kind - meaning that casting spells use both element of spell and kind of spell, rising them both, but also using them both as factor of successful spellcasting.

There are eight elements and six kinds of common magic in vanilla:

Elements	Kinds
Void	Destruction
Fire	Absorption
Electricity	Illusion
Soul	Transformation
Wind	Evocation
Ice	Restoration
Water	
Earth	

It is not required to make element-kind proportions equal (3 elements-3 kinds), however making them so is suggested. Non-equal variants give more freedom to player to choose, but also make no initial point.

You write common magic entries by using "elements" and/or "kinds" in such way:

```
"magic": {
    "elements": {
        "void": 2,
        "fire":  1
    },
    "kinds": {
        "destruction": 3
    }
}
```

♠ Path magic

Path magic is different from common one mostly because it conveys the idea that once chosen path is exclusive and do not allow for other paths to be taken. Additionally, choosing path magic weakens your spellcasting in common magic, at least with strongest spells offered.
In exchange, your dedication is rewarded with magic that can cross many boundaries, making you a powerful mage.

Since path magic is so exclusive, it is not used in race selection in vanilla, however the possibility to use it is still available.
While putting more than one path magic is not technically impossible, it may result in error in the future, as purpose of this magic type is to have exclusive choice.

There are seven magic paths for now in vanilla:

Path	Description
Dream magic	Path of wisdom of universes and oneironautics
Death magic	Cult of death and necromancy
Erd magic	Path of levitation and transmutation
Lunatri magic	Path of lunatris, ceremonial and nature-based
Ormath magic	Path set on communication with wind spirits
Terten magic	Path built on pestilence and plague
Keht magic	Path focused on blood sacrifices and vampirism

Similarly to common magic, path magic is created in *magic* section, but uses *path* as keyword:

```
"magic": {
    "path": {
        "dream_magic": 6
    }
}
```

IF. Names
Section dedicated to names offer you possibility to suggest player names for their character, based on the race they chosen.

Names are split by gender, using similar system to attributes/skills of vanilla genders being ID-agnostic.
Vanilla offers you *male*, *female* and *non-binary* genders to fill, however you do not need to use all of them. If you do not fill specific gender (or mod with more genders is added), ane "empty" gender will simply take all names from other genders available.

You can prevent this situation, if you need names being precisely only for genders they are designed for, by writing "*strict*": *true*.

```
"names": {
    "strict": true,
    "male": [
        "Smith",
        "Gregory"
    ]
},
```

Example above will suggest names of Smith and Gregory only to males, but if you remove "strict" keyword, it will offer those names to all genders not being covered.
It is important to notice that names are lists, using square brackets - [] - instead of curved ones - {}.

IG. Perks
Perks are special type of bonus given to specific race that allow them to gain some advantage, usually similar to genetics. Perks are rarely obtainable through quests, but not learnable in different way.

Since they are not polished enough, this part of tutorial will be expanded later.