Tab 1

followup thread can be found here: https://forums.frontier.co.uk/threads/discussion-thread-to-request-journal-changes-to-improve-3rd-party-apps.638119/

Bugs

Definition: Errors in journal entries or missing entries to keep a valid state of the game, where this was possible previously.

Care packages

Severity: High

Description:
There is no event for care packages. This is an issue for apps that report inventory counts and currently requires a relog. Please distinguish between materials that the commander was able to collect and materials that the commander was unable to collect (i.e. when the commander has insufficient inventory space). Only the shiplocker.json file is currently updated which contains only on foot materials.

Workaround: relog

Powerplay

Severity: Low

Description:
PowerplayMerits can log without a power defined. For example:

```
{ "timestamp":"2025-04-26T11:59:22Z", "event":"PowerplayMerits", "Power":"",
"MeritsGained":8, "TotalMerits":35912 }
```

Workaround: use power from earlier Powerplay/PowerplayJoin events

Shared mission rewards

Severity: High

Description:
Handing in a mission which has been shared with you should generate a MissionCompleted event just like any other mission but currently does not. This is an issue for apps which report credit balances and inventory counts and cannot currently be corrected without a relog.

Workaround: relog

Colonisation

Severity: Low

Description:
ColonisationConstructionDepot events are currently logged every 15 seconds while docked. Can cause issues with disk space due to continuous logging. Should only log upon docking and when it has changed.

Workaround: ignore duplicates

Exobiology

Severity: High

Description:
The `ScanOrganic` event has unreliable location information after the `Analyse` ScanType is reported. The issue is that this event is only generated after the animation is completed. If the player does their third scan and stows their scanner before the animation completes then the event isn’t generated. When the player disembarks at a different location and pulls their scanner back out, the animation resumes / completes and the event is generated with the wrong location information (system address and body).

Workaround:None

Crime and Punishment Faction Details

Severity: High

Description:
'RedeemVoucher', 'Bounty, and 'PayFines' events are missing faction details necessary for tracking the commander’s criminal state with factions and superpowers. (https://issues.frontierstore.net/issue-detail/29398) Example:
`{"event":"RedeemVoucher", "Type":"settlement", "Amount":4963, "Faction":"" }`
When redeeming bounty vouchers from Powerplay contact:
`{ "timestamp":"2025-02-03T08:38:48Z", "event":"RedeemVoucher", "Type":"bounty", "Amount":342789, "Factions":[ { "Faction":"", "Amount":322927 }, { "Faction":"", "Amount":19862 } ] }`

Workaround: None

FactionKillBond

Severity: Moderate

Description:

The `FactionKillbond` event significantly under-reports both ship combat and on foot combat bond income (reporting only 25-33% of actual earnings - apparently an amount equal to the pre-buff earnings from long ago - see https://forums.frontier.co.uk/threads/game-balancing-pt-3.560418/). Actual expected earnings cannot be tracked accurately.

Workaround: None

Changes

Settlement Layout

Type
Additional field(s) which is/are not present in UI

Issue
The `ApproachSettlement` and `Docked` events don't provide details on the settlement/station layout. This can somewhat be deduced from the landing pads, but some settlements have the same configurations.

Potential Resolution
Add the settlement/station layout to the `ApproachSettlement` and `Docked` events.

Benefit
Settlement layouts can help 3rd party applications provide information to the player about the specific settlement layout. It can help players locate specific settlement layouts.

Cargo Status

Type
Additional field(s) which is/are present in UI

Issue
The `CargoTransfer` event currently lacks information about the legal status of the cargo being transferred. This missing data makes it difficult for third-party tools to track and monitor the movement of illegal cargo in a commander's inventories.

Potential Resolution
Enhance the `CargoTransfer` event to include a field indicating the legal status of the cargo (e.g., legal, illegal, or restricted). This would allow tools and players to accurately track the legality of cargo being moved between inventories.

Benefit
Adding legal status information to the `CargoTransfer` event would enable third-party apps to more effectively monitor contraband and illegal cargo, improving the ability to track legal

violations and enhancing gameplay features related to cargo management, piracy, and law enforcement.

Unusual Mission Givers

Type
Missing existing event

Issue
When missions are accepted from NPCs walking around in a settlement or through in-ship messages, no `MissionAccepted` journal event is written, even though `MissionRedirected` and `MissionCompleted` events are correctly logged. This omission makes it difficult for third-party apps to track missions from the point of acceptance, impacting mission management and tracking.

Potential Resolution
Ensure that a `MissionAccepted` event is written when missions are accepted from NPCs in settlements or via in-ship messages. This would create a complete and consistent event flow for mission acceptance, redirection, and completion.

Benefit
By including the `MissionAccepted` event for these mission sources, third-party tools can more reliably track the lifecycle of missions, improving the user experience for players and enhancing mission management features in external applications.

Mission Expiration

Type
Additional field(s) which is/are present in UI

Issue
When an Odyssey on-foot mission is redirected, the mission expiration timer can change (typically by +1 day), but this change is not reflected in the player journal. This lack of updated expiration information makes it difficult for third-party apps to accurately track active missions and their timers.

Potential Resolution
Update the `MissionRedirected` event to include any changes to the mission expiration timer. This would ensure that the new expiration time is accurately recorded in the journal, allowing external apps to track mission deadlines properly.

Benefit
Including the updated expiration time in the `MissionRedirected` event would improve the reliability and accuracy of mission tracking in third-party tools, ensuring that players have up-to-date information on their active missions. This would reduce confusion and help players manage their mission deadlines more effectively.

Mission Locations

Type
Additional field(s) which is/are present in UI
Missing existing event

Issue
When a mission is modified by a wrinkle—such as a change in the mission target's location—no corresponding journal event (e.g., `MissionRedirected`) is written to reflect the update. This lack of information creates problems for apps that track active missions, as they are unable to detect or respond to such changes.

Potential Resolution
Introduce the writing of relevant journal events when a mission is modified by a wrinkle. Additionally, consider adding `DestinationBody` and `DestinationStation` properties to appropriate mission-related events to explicitly convey location updates.

Benefit
Enhancing journal events in this way would allow third-party apps to more accurately track and reflect mission changes, improving reliability and user experience for players relying on external mission tracking tools.

Mission Boards

Type
New file

Issue
Currently, there is no record of available missions when viewing the mission board. This limits the ability of external tools to analyze mission options in real time.

Potential Resolution
Log available missions to `missionboard.json` or emit a dedicated journal event when the mission board is viewed. This should include key details such as mission descriptions, rewards, factions, destinations, and mission types.

Benefit
Providing this data would enable third-party tools to evaluate and recommend missions tailored to the commander's current ship loadout and preferred playstyle. This enhances gameplay by supporting more informed decision-making and personalized mission selection.

Module identifiers

Type
Additional field(s) which is/are not present in UI

Issue
Ships have a ShipID, but modules lack a ModuleID. This makes it near impossible to reliably track the location of modules.

Potential resolution
Add a ModuleID to all events containing modules.

Benefit
A unique identifier in the ship loadout, module transfer, module engineering, etc. events would allow better tracking of the module location, whether or not a specific engineered module is equipped on a ship, and whether a specific module is considered "hot" because it was used to commit a crime.

Shipyard and Outfitting

Type
Additional field(s) which is/are present in UI

Issue
It is not possible to reliably calculate the original price from a discounted/marked up price that has been discounted/marked up and rounded multiple times. For example ShinDez will apply: 10% discount, round, 2.5% elite discount, round. It is more useful to know the base price and what discounts were applied as we can reliably calculate it the other way around.

Potential resolution
Include non-discounted/marked up prices for ships and modules. Additionally to or instead of the discounted price, an array with discount values. [10.0, 2.5]

Benefit
Accurate data for tools that use price information, like ship builders. Accurate prices independent of personal discounts to be shared with others.

Complete ships/modules inventory

Type
New file

Issue
It is hard to get a complete inventory of all the ships and modules. It requires scraping of all journal files which are also potentially incomplete.

Potential resolution
Could be written to separate files on startup. Should contain all information like unique identifiers, engineering and location. Could be an expansion of the StoredShips event and a new StoredModules event for modules that are not equipped. The StoredShips should also indicate the slots the modules are equipped in, very much like the Loadout event.

Benefit
Reliable data for tools that aid in managing inventory, helps locate specific modules. Makes it possible to easily manage ship builds.

Complete loadouts/suits/weapons inventory

Type
New file

Issue
It is hard to get a complete inventory of all loadouts, suits and weapons. It requires scraping of all journal files which are also potentially incomplete. Capi showed signs of this being implemented in some capacity, but shows NYI(Not Yet Implemented?) to this date.

Potential resolution
Could be written to a seperate file on startup. Should contain all information like identifiers and engineering.

Benefit
Reliable data for tools that aid in managing inventory and design suit and weapon loadouts.

StoredModules

Type
Additional field(s) which is/are present in UI

Issue
The `StoredModules` event currently provides information about engineered modules but omits details about the experimental effects applied to those modules. Without this information, third-party tools and players cannot fully track or analyze the experimental modifications made to their modules.

Potential Resolution
Enhance the `StoredModules` event to include the experimental effects applied to engineered modules. This would provide a complete picture of the module's configuration, including both the engineering and experimental modifications.

Benefit
Including experimental effects in the `StoredModules` event would allow third-party tools to provide more comprehensive tracking of module upgrades, improving player experience and ensuring that all aspects of module engineering are accounted for when analyzing or managing ships.

Module Status

Type
Repeat existing event
New event

Issue
The `ModulesInfo.json` file is only generated when the player opens the `Modules` sub-panel and is not updated when changes are made—such as adjusting priorities or

toggling module states. As a result, it cannot be used to reliably track the current status of player modules. Additionally, the file does not include any information on module health or damage levels, making it impossible for tools to monitor module condition or respond to damage events.

Potential Resolution
Update `ModulesInfo.json` whenever module states are changed or at least upon exiting the sub-panel. Include module health and damage information in the file. Alternatively, introduce a new `ModuleDamage` journal event, similar to `HullDamage`, which logs damage to individual modules at meaningful thresholds (e.g., every 20%).

Benefit
These improvements would enable third-party tools to accurately monitor and display module status, including health and configuration changes. This would support real-time diagnostics, damage tracking, and alerts, enhancing player awareness and control over ship systems.

Pre-engineered modules

Type
Additional field(s) which is/are present in UI

Issue
The game identifies legacy and pre-engineered modules in the UI. These modules are harder to identify in the journal because the journal entries are still from the legacy system. The game clearly knows these modules are different.

Potential resolution
A boolean flag on the modules to indicate legacy or pre-engineered status will help 3rd party apps identify them properly. Can be omitted for regular modules.

Benefit
This will allow 3rd party apps to reliably detect legacy and pre-engineered modules.

Pre-built ship modules

Type
Additional field(s) which is/are present in UI

Issue
Pre-built ship modules are not identifiable in the journal. The game identifies pre-built shipmodules in the UI. Pre-built shipmodules can not be stored and exchanged between ships.

Potential resolution

A boolean flag on the modules to indicate pre-built shipmodules will help 3rd party apps identify them properly.

Benefit
This will allow 3rd party apps to reliably detect pre-built shipmodules.

Codex entry locations

Type
Additional field(s) which is/are present in UI

Issue
The `CodexEntry` journal event lacks precise location data such as `BodyName`, making it difficult to accurately track where codex discoveries occur. As shown in the EDDN schema documentation, third-party tools must use indirect methods to infer location, which is error-prone and inconsistent.

Potential Resolution
Include a `BodyName` field (and potentially other location-relevant data) directly in the `CodexEntry` event. This would provide an explicit reference to the planetary body or location where the codex discovery was made.

Benefit
Adding `BodyName` would greatly simplify and improve the accuracy of codex tracking for third-party tools, enabling more reliable data collection and analysis for explorers and community science initiatives.

Trader/broker type

Type
Additional field(s) which is/are present in UI

Issue
The `ApproachSettlement`, `Docked`, `CarrierJump` and `Location` journal event lacks precise types for traders and brokers and only lists whether a trader or broker is present at the station.

Potential Resolution
Replace the existing stationservice listed in the events with more specific versions of it. Instead of materialtrader and techBroker use materialtraderraw and techBrokerGuardian etc.

Benefit
Allows accurate data collection and allows 3rd party apps to direct players in the right direction.

Market Data

Type
Repeat existing event

Issue
Market data is only recorded when the player enters the market. If stock levels change—either due to player purchases or external influences—these changes are not captured in real time. As a result, third-party tools display outdated stock information, which can mislead other players relying on that data.

Potential Resolution
Add a journal event that logs market data changes whenever stock or demand is updated, whether by player actions or external game state changes. This event could be triggered during transactions or periodically when market conditions fluctuate.

Benefit
Recording the state of the market at the market close would significantly improve the accuracy of third-party market data tools. This would support better trade planning and reduce misinformation, benefiting the player community with more dependable economic insights.

Permits

Type
New event

Issue
To direct players around the bubble it can be important to know whether a system is possible to visit due to a permit lock, for example the engineer in Sol.

Potential Resolution
A new event that lists permits, written at startup and when a permit is unlocked.

Benefit
Allows 3rd party apps to give accurate directions to systems and stations and inform players on hidden unlock requirements for engineers.

FSDTarget

Type
Additional field(s) which is/are present in UI

Issue
The `FsdTarget` event currently does not include coordinates (`StarPos`) for the target star system. This missing data makes it challenging for third-party apps and tools to calculate the commander’s position relative to the target system and accurately gauge distances for travel and exploration.

Potential Resolution
Enhance the `FsdTarget` event to include the `StarPos` coordinates of the target star system. This would provide the necessary data to determine the relative position and distance to the target system.

Benefit
Including `StarPos` coordinates in the `FsdTarget` event would improve navigation oriented third-party apps by allowing them to accurately calculate travel distances. It would also enhance the player experience by providing more detailed positional data for journey planning and exploration.

FSS

Type
New event

Issue
The `FSSSignalDiscovered` and `FSSBodySignals` events are logged completely only the first time an object or signal is scanned, but subsequent scans of the same object do not trigger a new event. This makes it difficult for third-party apps to report updated scan data for objects that have already been scanned, as the event is not re-logged upon further scans.

Potential Resolution
Introduce additional events `FSSSignalRescanned` and `FSSBodyRescanned` that are triggered whenever an object is scanned again after its initial scan. This event should log the updated scan values, allowing third-party apps to track and report the scan data more accurately.

Benefit
This additional event would allow third-party apps to keep up-to-date records of scanned objects, improving the accuracy of reports, especially when tracking objects with variable scan values. Players and developers would benefit from enhanced tracking capabilities for system exploration, making it easier to monitor changes in scan data over time. Third-party apps can provide meaningful feedback to the player upon rescanning an object, for example scan value.

FSSSignalDiscovered

Type
Additional field(s) which is/are present in UI

Issue
Currently, there is no journal data for the distance from the entry star to objects such as signal sources, bodies, installations, or conflict zones. Additionally, ownership information for various objects, such as installations, USSs, High Grade Emissions (HGEs), and conflict zones, is not included, even though this information is visible in the system map/UI. The lack of this data limits the ability of third-party tools to track and analyze objects and their properties effectively.

Potential Resolution
Enhance journal events to include the following information:

1. **DistanceFromArrivalLS**: Include the distance from the entry star for objects such as signal sources, bodies, and installations.

2. **Body Information**: Include information about the body the signal is orbiting.

3. **Ownership Information**: Provide faction ownership details for installations, USSs, and other signal sources. Include ownership information about the participants in conflict zones.

Benefit
Including these data points in the journal would greatly enhance third-party apps and tools, allowing for better tracking and analysis of objects and events in the game. Players would benefit from more detailed insights into the location and ownership of various in-game objects, improving decision-making and strategic planning. It would also provide greater transparency for conflict zone tracking and more immersive data for exploration and discovery.

USS Navbeaconscan

Type
Additional field(s) which is/are present in UI

Issue
The `NavBeaconScan` event currently lacks the timer value for the Emission Signal Sources, even though this information is visible in-game when targeting the Signal Source. The absence of this timer value from the journal makes it harder for third-party tools to accurately track Signal Sources, especially when compared to the data provided by an FSS scan that includes the timer.

Potential Resolution
Include the timer value for the Signal Source in the `NavBeaconScan` event, similar to how it is provided in the FSS scan. This would ensure that the timer information is consistently available across different scanning methods.

Benefit
Including the USS timer value in the `NavBeaconScan` event would allow third-party apps to better track Signal Sources, improving gameplay features related to exploration and signal source management.

Dataports/Lockers

Type
New event

Issue
Currently, there are no journal events that specify what materials are found in lockers or dataports, nor where (e.g., building, device, or POI) they are located. This lack of detail hinders third-party tools from providing precise tracking of material sources and makes it difficult to gather meaningful statistics on material distribution.

Potential Resolution
Introduce new journal events that are triggered when a locker or dataport is first accessed. These events should list the materials found and identify the associated location or container (e.g., building type, device, or POI name).

Benefit
Providing this information would allow third-party tools to assist players in locating desired materials more effectively and enable the community to build reliable datasets on drop rates and material locations. This improves both individual gameplay efficiency and broader knowledge sharing.

Pinned blueprints

Type
New event

Issue
There is currently no journal event that records which blueprints a player has pinned at engineers. As a result, third-party tools cannot track pinned blueprint status unless players manually update or synchronize this data.

Potential Resolution
Add a journal event written at game startup that lists all currently pinned blueprints, including both the engineer and the specific blueprint. Additionally, log a separate event whenever a new blueprint is pinned at an engineer.

Benefit
These additions would allow third-party tools to maintain an up-to-date view of a player's pinned blueprints, improving engineering planning. It would also reduce manual effort and help avoid duplicate or missed blueprint goals.

Odyssey ground conflict zone information

Type
Additional field(s) which is/are present in UI

Issue
Players looking to participate in ground conflict zones (GCZs)—whether to earn high payouts or to unlock achievements like *Hero Ferrari*—can currently face a tedious and inefficient process to locate a specific type of conflict zone. To determine conflict intensity for example, a player must:

1. Locate a system with a war,
2. Check if it has landable bodies,
3. Travel to the system,
4. Land at a port,
5. Disembark and visit Frontline Solutions.
   This is particularly burdensome for players using their own ships, as they must repeat the process for each candidate system, often needing to backtrack if the system lacks viable GCZs.

Potential Resolution
Include ground conflict zone (GCZ) data directly in `FSDJump`, `Location`, and `CarrierJump` events within the `Conflicts[x]` map. This data should contain:

- Settlement name
- Faction names (attacker and defender)
- Conflict intensity (Low, Medium, High)
- Distance from arrival in LS
- Optionally, associated planetary body information

Benefit
Adding this data to jump-related events would allow players and third-party tools to identify high- or low-intensity GCZs without having to physically visit stations in the system. This would streamline gameplay and support goal-oriented exploration (e.g., community goals or engineer unlocks).

Status

Type
Additional field(s) which is/are present in UI

Issue
The current `Status.json` output lacks several key indicators that would help third-party tools and players better understand their real-time status. Missing data includes important ship status effects (like caustic damage or shutdowns), personal suit stats (like energy levels), positional and environmental data (like gravity), and various GUI states. Additionally, certain flags or fields are inconsistently populated depending on whether the player is on foot, in a ship, or in an SRV—limiting reliable context awareness.

Potential Resolution
Enhance `Status.json` by including the following:

<u>Ship-Related Additions:</u>

- Flags for ship status effects (e.g., caustic damage)
- A flag for shutdown state (due to reboot or Thargoid shutdown field)
- Emergency oxygen status or remaining oxygen in milliseconds
- Flag for rotational correction status

On-Foot Additions:

- Suit energy level (complementing oxygen and health)
- Proper tracking of: shields, night vision, lights, planet radius, altitude, landed status, landing gear down, and danger level (`InDanger`)

SRV Additions:

- Proper setting of the "Hud in Analysis Mode" flag

General Info:

- Include gravity in `Status.json` positional information when near a surface (not just when on foot)
- Notoriety level

GUI State Enhancements:

- Expand `GuiFocus` values to include:
  - Carrier Management
  - Carrier Galaxy Map and System Map variants
  - Apex Galaxy Map and System Map variants
  - External Camera
- Update `GuiFocus` while viewing maps on foot

Destination Field Enhancements:

- Include `StarPos` (coordinates)
- Include `MarketID` if available

Benefit
These additions would significantly increase the fidelity of third-party tools and telemetry systems, enabling better status tracking, UI automation, HUD overlays, and safety alerts. They also reduce the need for inference or workarounds, making tool development more robust and improving quality of life for both developers and players.

Crime and Punishment

Type
Additional field(s) which is/are present in UI
New file

Issue
Currently, there is no centralized way to track the fines, bounties, vouchers, and notoriety affecting a player. This makes it difficult for third-party tools to accurately aggregate and report a player's legal status across different factions and superpowers. Additionally, certain events, like `PayBounties` and `CommitCrime`, lack the necessary data to track the

conversion of minor faction bounties to superpower bounties, and `Notoriety` is not accessible.

Potential Resolution

1. **Crime Aggregation:**
    - Introduce a `crime.json` file or an event at startup summarizing:
        - Fines and bounties owed to various factions and superpowers
        - Vouchers awarded by factions and superpowers
        - Current notoriety level
    - This file would serve as a central resource for tracking legal status.

2. **Notoriety Tracking:**
    - Add `Notoriety` to `crime.json`, `status.json`, or provide it as part of a new startup event.
    - Introduce a `NotorietyChange` event to track any changes in notoriety over time.

3. **PayBounties Event Update:**
    - Update the `PayBounties` event to list both the minor faction and, when relevant, the superpower receiving the bounty payment. This would allow external apps to track fines and bounties accurately by faction.

4. **Tracking Bounty Conversion:**
    - Add an event to document when minor faction bounties are converted into superpower bounties. This event should include:
        - The superpower
        - The minor factions whos fines and bounties are being aggregated into the superpower bounty
        - The credit amount of the new superpower bounty
    - Alternatively, add the victim's allegiance to the `CommitCrime` event so that players and tools can track which superpowers are involved in crimes and identify when bounty conversions take place.

Benefit
These improvements would allow third-party apps and tools to more effectively track and manage the player's legal status, making it easier to track fines, bounties, notoriety, and crime-related interactions with factions and superpowers. They would also help developers build more accurate crime tracking systems, improve user awareness, and ensure that all relevant data is accessible for analysis and decision-making.

Fleet Carriers

Type
Additional field(s) which is/are present in UI
Repeat existing event
New event

Issue
The `CarrierStats` event is currently only generated when the player accesses the "carrier management" screen, which means this information is unavailable when the player loads their game. Additionally, course plotting events for carriers lack sufficient journal feedback for situations where the commander is not aboard. The `CarrierJump` event also omits critical station properties when the commander is disembarked and on foot, making it impossible to distinguish between the commander's carrier and another player's.

Potential Resolution

1. Generate `CarrierStats` events at the point of loadgame to provide carrier data right when the game starts, allowing tools to track carrier stats without needing to access the management screen.

2. Enhance the journal feedback for carrier course plotting to include more detailed events such as:

    - Plotting failed
    - Jump started
    - Jump complete
    - Cooldown complete

3. Modify the `CarrierJump` event to include station properties even when the commander is on foot, ensuring that the event accurately identifies whether the jump is from the commander's carrier or another player's.

Benefit
These improvements would provide more robust tracking and interaction with carriers, enhancing both player experience and third-party tool functionality. By allowing access to carrier stats at loadgame, improving carrier course plotting feedback, and ensuring full event details even when disembarked, players and developers would have more reliable, comprehensive data for managing and monitoring carriers.

Exobiology

Type
Additional field(s) which is/are present in UI

Issue
The `ScanOrganic` event currently lacks location-specific data such as the planetary body name (`BodyName`) and the player's latitude and longitude. This makes it difficult for third-party tools to accurately track the specific location of organic scans, particularly when gathering detailed exploration data (science!) or analyzing player activities.

Potential Resolution
Modify the `ScanOrganic` event to include the following additional information:

- `BodyName` (name of the planetary body where the scan occurs)
- Player's current `Latitude` and `Longitude` coordinates on the body

Benefit
Including this data in the `ScanOrganic` event would allow third-party tools to more accurately map organic scan locations, enhancing exploration and data tracking capabilities. It would also improve the utility of tools focused on exploration, giving players more detailed insights into where specific organic lifeforms were found and tracked.

Integrity

Files from any commander

There are several json files used to reduce the amount logged to the journal. Nowadays a lot of commanders have multiple accounts.
There is no way to see who the files belong to. It would be nice if these files included the commander's FID or were stored in a subfolder for each commander, using the FID as an identifier.
This will increase the integrity of the data. Even the journals can be stored in subfolders so a folder only contains data related to that specific account.

State Files

The additional json files are used to write bigger events and reduce the space consumption in the journal file itself. They come close to being state files and it would be nice if they actually were! It would solve issues like the missing care packages and prevent them in the future. It removes the need for 3rd party apps to construct state from the journals themselves. It would remove the need to write the bigger journal events at the start of each journal and would likely simplify the journal writing code for FDev, since a lot of events are state related. A lot of the suggestions made above could also be written as state files instead. What should the state files contain?
- ships with full loadout and engineering information and locations
- suit/weapons with full mods and engineering information
- stored modules with engineering information and locations
- player stats (location, money, permits, rank progression, techbroker unlocks)
- current inventory materials/commodities in backpack/ship/fleetcarrier
- fleet carrier location and open orders(buy/sell)
- crime and punishment (bounty vouchers, bounties, fines, with information about the factions and superpowers tied to those values; notoriety information)