Note: This piece of documentation is still considered a work-in-progress and is rapidly evolving. Please consult @Australis#6709 on Discord for more information about the API until then.

Config Data Structure:

Config

- **Alerts** (**Note:** Not to be confused with Events. Although both alerts and events are displayed in the UI instead of in PM's, alerts are typically there to inform the player of important decisions, whereas events serve as a catch-all for everything else).
    - **Alert Category**
        - **name*** (String): Sets the human readable name of the given alert.
        - **description*** (String): Sets the description of the given alert upon popping up.

        - **btn_<name>**
            - **name*** (String): Sets the human-readable name of the button.

            - **ai_chance*** (Integer): Determines the chance of the AI resolving the alert with this option upon expiry. (**Note:** This part of the API will likely be reworked in the future due to lack of flexibility to include dynamically weighted values and limits). Defaults to 100 for all options if no AI chances are specified. If other AI chances are specified, this defaults to 0.
            - **effect*** (Effect Scope): These effects are always applied upon being taken by either the AI or a human player.

- **Buildings**
    - **Building Category**
        - **name*** (String): Sets the name of the building category in UI menus.
        - **always_display*** (Boolean): Determines whether or not the building category should always be displayed in UI menus.
        - **disable_slots*** (Boolean): Disables category-wide building slots if true, this can later be reduced down to the Building scope, or used to remove build cap limits.
        - **order*** (Number): Determines the order in which the building category appears in UI menus.
        - **Building**
            - **name** (String)
            - **singular** (String)
            - **aliases*** (String/Array)
            - **icon*** (Emoji String)

            - **construction_turns*** (Number): Determines how many turns the building should take to construct by default.

        - **cost*** (Good List Scope): What the building costs to construct.
        - **houses*** (Integer): Determines how much is added towards total housing capacity in the province by this building.
        - **maintenance*** (Good List Scope): What the building costs to maintain each turn; maintenance affects production by default.
        - **modifiers*** (Modifiers Scope): Determines what modifiers are applied or taken away when the building is built/destroyed.
        - **produces*** (Good List Scope): What the building produces each turn.
        - **separate_building_slots*** (Boolean): Whether or not the building uses individual building slots as opposed to one for the entire building category.
        - **supply_limit*** (Integer): Determines what the building adds/deducts from local supply within its range.
        - **unlimited_slots*** (Boolean): Removes all building slot caps if true.
- **Defines**
    - **Combat**
        - **access_territory_modifier** (Percentage): Multiplies the base amount of supply by this percentage if a user's army is standing in a province to which it is either allied, vassalised, or has military access.
        - **anti_aircraft_effectiveness** (Percentage): Multiplies the AP damage presented by ack-ack guns in a city during air raids.
        - **army_speed** (Integer): Sets how fast the army travel speed should be in provinces per turn for armies.
        - **base_attrition_rate** (Inverse Percentage): Determines the base rate of attrition for all units at game start if attrition is triggered.
        - **base_mobilisation_impact** (Effect Scope): Determines the types of modifiers applied to a country if mobilisation is active; this is by default production_efficiency, tax_efficiency, and rgo_throughput. When a nation demobilises, these modifiers are attempted to be undone by removing the National Modifier responsible for them.
        - **base_mobilisation_time** (Integer): Determines how many turns it should take to fully mobilise at first.
        - **base_mobilisation_size** (Percentage): What the default percentage of mobilised workers should be.
        - **base_supply_limit** (Integer): Multiplied by 1000. Determines how many thousands of soldiers a province can supply by itself without any supporting infrastructure.
        - **base_transfer_time** (Integer): Determines how many turns all shipments should take at the least to simulate the packing and gathering of goods. Can be set to '0' or deleted in order to be disabled.
        - **blockade_challenge_limit** (Integer): Determines how many times a single fleet can challenge an enemy blockade on the same turn.

- **colonisation_speed** (Integer): Determines how fast colonists can travel per turn by default without any additional modifiers in provinces.
- **friendly_territory_modifier** (Percentage): Multiplies the base amount of supply by this percentage if a user's army is standing in a province with an accepted culture.
- **hostile_territory_modifier** (Percentage): Multiplies the base amount of supply by this percentage if a user's army is standing in a province to which it is at war with. Also applies to uncolonised land. Overridden by friendly_territory_modifier.
- **infrastructure_range** (Integer): Impacts the overall distance by which infrastructure has a fixed impact on supply, or the radius of provinces.
- **interception_range** (Integer): Impacts the range at which air wings are allowed to intercept enemy bombing raids.
- **max_army_creation_limit** (Integer): Determines how many armies a nation is allowed to create at once. This cannot be modified by any other modifiers.
- **max_army_limit** (Integer): Determines the maximum amount of armies a nation is allowed to have at once.
- **max_army_speed** (Integer): Can be set to '0' or deleted in order to be disabled, determines the maximum army speed in provinces per turn.
- **max_colonisation_speed** (Integer): Can be set to '0' or deleted in order to be disabled, determines the maximum colonisation speed in provinces per turn.
- **max_shipment_speed** (Integer): Can be set to '0' or deleted in order to be disabled, determines the maximum shipment speed in provinces per turn.
- **mobilisation_cooldown** (Integer): Determines how long countries must wait between mobilisation and demobilisation.
- **mobilise_unit** (String): The default unit type for mobilisation. Invalid unit types are presented as none.
- **peacetime_mobilisation_penalty** (Number): Determines the amount of infamy gained per turn if a country mobilises during peacetime.
- **shipment_time** (Number): Determines the base transfer speed for manual shipments of goods in provinces per turn.
- **war_exhaustion_blockade_limit** (Percentage): Determines the maximum warscore penalty for being blockaded overtime.
- **war_exhaustion_blockade_rate** (Percentage): Determines how fast war exhaustion should tick up if a user is blockaded by default. This can be reduced by the blockade_efficiency modifier.
- **war_exhaustion_mobilisation_rate** (Percentage): Ticks up war exhaustion by this amount per turn if the country is mobilised.
- **war_exhaustion_tickdown_rate** (Percentage): The base speed for war exhaustion tickdown if a user is at peace.

- **Common**

    - **starting_kit** (Effect Scope): Applies effects inside of this scope whenever triggered, which is whenever a nation is first founded or settled.
    - **starting_year** (Integer): Determines the starting year of the scenario. Decimals in starting_year can also determine the season of the start date.
- **Diplomacy** (**Note:** This API is very likely to be refactored in the future to be more dynamic and allow for custom-scripted diplomatic actions, but can currently be modded only through defines).
    - **annex_cost** (Integer): Amount of Political Capital needed to attempt the annexation of another country.
    - **break_alliance_cost** (Integer): Amount of Political Capital required to break off an alliance.
    - **cede_city_limit** (Integer): Determines how many provinces containing cities a country is allowed to cede per turn; counts towards cede_province_limit.
    - **cede_province_limit** (Integer): Determines how many provinces a country is allowed to cede per turn.
    - **declare_rival_cost** (Integer): Amount of Political Capital required to declare a new rival.
    - **decrease_relations_cost** (Integer): Amount of Political Capital needed to decrease relations over time. One-time cost.
    - **form_alliance_cost** (Integer): Amount of Political Capital needed to attempt the formation of an alliance with another country.
    - **guarantee_independence_cost** (Integer): Determines the Political Capital cost of guaranteeing the sovereignty of another country. One-time cost.
    - **improve_relations_cost** (Integer): Amount of Political Capital needed to improve relations over time. One-time cost.
    - **justify_wargoal_cost** (Integer): Determines the amount of Political Capital that is required and consumed when justifying a wargoal.
    - **justify_wargoal_time** (Integer): The base amount of turns needed to justify a wargoal.
    - **request_military_access_cost** (Integer): Determines the amount of Political Capital required for requesting military access from another player.
    - **rival_slots** (Integer): Determines the base amount of rival slots made available to a country at the start of the game. Will be moved to modifiers in future.
    - **sign_non_aggression_pact_cost** (Integer): Determines the Political Capital cost of attempting to sign a non-aggression pact with another country.
    - **vassalise_cost** (Integer): Amount of Political Capital needed to attempt the vassalisation of another country.
- **Economy**

    - **city_resources** (Goods Scope): Determines what resources are required for founding a new city.
    - **city_unlock_caps** (Integer/Array): The various amounts of provinces at which the city unlock cap is raised. This mechanic can be disabled either by deleting the variable entirely, or leaving it empty.
    - **construction_turns** (Integer): Determines the default number of construction turns per city.
    - **money_per_action** (Integer): How much money should be gained from each action assuming an effective tax rate of 100%.
    - **resource_base_stock** (Integer): How much of each good should be placed in the World Market at the start of the game.
    - **resource_init_max_percentile** (Percentage): Determines the maximum percentage of Shipment Capacity that new countries for the first 10 turns can use on buying or selling goods from the World Market. (**Note:** This part of the API will likely be reworked in the future).
    - **resource_max_percentile** (Percentage): Determines the maximum percentage of Shipment Capacity that countries after the first 10 turns can use on buying or selling goods from the World Market.
    - **resource_min_buy_price** (Integer): Sets the minimum purchasing price for a resource.
    - **resource_min_sell_price** (Integer): Sets the minimum selling price for a resource.
    - **rgo_modifier** (Percentage): Sets the amplifier for RGO modifier effects gained from settling provinces/founding cities.
    - **shipment_speed** (Integer): Determines the base speed of a manual shipment in provinces per turn.
    - **starting_actions** (Integer): Determines the base number of actions received at the start of the game in addition to how many actions a user receives per turn. Can be overridden by starting_kit.
    - **urbanisation_cost** (Integer): Sets the base Political Capital cost that is required and consumed for a country to urbanise a city (or add one slot to a building category of its choice).
    - **urbanisation_cost_percentile_growth** (Percentage): Determines how fast the Political Capital cost for further urbanisation of a province should grow exponentially depending on the level of development.
- **Map**
    - **map_background** (String, File): Determines the local file path within the map folder to the background image for the map.
    - **map_definition** (String, SVG File): Determines the local SVG file path within the map folder to the definition file for the map. This contains all the provinces as separate path elements within an SVG.
    - **map_label_coords** (Array, <X, Y>): Specifies the positioning of country labels on the map, or the general key for the map. This is a required field as the key should generally be as unobtrusive as possible.

    - **map_name*** (String): The specified human-readable name of the map that can be used for choosing which map to play on initially. (**Note:** This define variable currently has no actual effect in-game).
    - **map_overlay** (String, File): The local file path within the map folder for the Province ID overlay on top of the map. Can be left blank, but be warned, as no Province ID's will then show up by default.
    - **map_resolution** (Array, <Width, Height>): Specifies both the width and height of the map in pixels.
    - **map_type** (String): Determines the name of the map. Various localisation dynamics and AI scopes can be applied based on this value.
    - **province_amount** (Integer): The total amount of provinces in the map or the maximum upper bound of Province ID's.
- **Politics**
    - **accepted_culture_cost** (Integer): Determines the cost in Political Capital for accepting a culture.
    - **accepted_culture_maintenance_cost** (Integer): Determines the base cost in Political Capital per turn for each accepted culture.
    - **assimilate_cost** (Integer): Sets the Political Capital cost of assimilating each individual province.
    - **assimilate_fiscal_cost** (Goods Scope): Determines the resources that are required towards
    - **coup_cost** (Integer): Determines the cost in Political Capital for orchestrating a coup.
    - **infamy_limit** (Integer): Determines the maximum infamy limit for all nations beyond which negative modifiers are applied. (**Note:** This part of the API will likely be reworked in the future).
    - **infamy_loss** (Number): Determines the base tickdown rate per turn for infamy when no actions of perfidy are being taken.
    - **max_stab_boost** (Percentage): Determines the maximum percentage a country can artificially boost their stability by. Can be set to '0' to disable stability boosts, or removed altogether for no limit on boosting.
    - **move_capital_cost** (Integer): Determines the base Political Capital cost for moving a player's capital to a different city.
    - **political_capital_gain** (Integer): Sets the minimum base amount of Political Capital gained each turn by players.
    - **reform_cost** (Integer): The Political Capital cost of attempting a reform.
    - **revolt_threshold** (Percentage): Determines how low stability needs to be before the country is capable of revolting to a different government.
    - **stability_cost** (Integer): The Political Capital cost for boosting stability by 10 per cent.
- **Technology**
    - **additional_research** (Integer): Determines the base research for all nations up to a certain research_cost. Can be set to '0' to be disabled.

        - **ahead_of_time** (Integer/Array<min_year, max_year, years_ahead_of_time_penalty>)
            - min_year (Integer): Determines the minimum year for the given range.
            - max_year (Integer): Determines the maximum year for the given range.
            - years_ahead_of_time_penalty: The tech penalty doubles for this many years that the tech is ahead of time within the specified range.
        - **max_knowledge_investment** (Percentage): Determines how much of a tech can be researched each turn at maximum. Can be set to '1' to be disabled. Related to the Knowledge Investment mechanic.
- **Events**
    - **Event Category**
        - **name*** (String): Determines the human readable name of the event.
        - **description*** (String): Gives the description of the event when printed out.
        - **image*** (String URL): Any image string to display as the event image.

        - **trigger*** (If Scope): Triggers the event if all conditions in the scope are met.
        - **duration*** (Integer): Determines how long the event lasts for before the AI decides to intervene and auto-resolve the event.
        - **immediate*** (Effect Scope): Determines any immediately triggered effects when the event fires.

        - **option_<name>** (Option Scope): Creates a new option for the given event.
            - **name*** (String): Sets the human-readable name of the given option.

            - **ai_chance*** (Integer): Determines the chance of the AI resolving the event with this option upon expiry. (**Note:** This part of the API will likely be reworked in the future due to lack of flexibility to include dynamically weighted values and limits). Defaults to 100 for all options if no AI chances are specified. If other AI chances are specified, this defaults to 0.
            - **effect*** (Effect Scope): The effects applied to the target nation upon taking this option.
            - **limit*** (If Scope): The option only shows up if all conditions in this block resolve to true.

- **Goods**
    - **Goods Category**
        - **name*** (String): Sets the name of the goods category in UI menus.

        - **order*** (Number): Determines the order in which the good category appears in UI menus.
        - **Good**
            - **name*** (String): Sets the human-readable name of the good.
            - **aliases*** (String/Array): Determines the other references to the good.
            - **icon*** (Emoji String): Sets the GFX icon for the good.

            - **_action_chance** (Integer): Creates a new manual action command for the good if one doesn't exist already. Ex: $quarry, $mine, etc.
            - **buy_price*** (Integer): Sets the buy price of the good in the market.
            - **default_stock*** (Integer): Sets the default stock of the good in the market.
            - **doesnt_stack*** (Boolean): If true, goods of this type are reset to zero at the beginning of each turn.
            - **is_cp** (Boolean): If true, the type of the good is set to 'Crafting Point' and is parsed as such.
            - **pop_demand*** (Pop Demand Scope)
            - **research_good*** (Boolean): Determines
            - **sell_price*** (Integer): Sets the sell price of the good in the market.

    - **hidden**: Goods in this scope do not show up in any interface/UI commands.

- **Governments**
    - **Government**
        - **name** (String): Sets the human-readable name of the government.
        - **adjective** (String): Determines the adjective of the ideology type (e.g. communists, constitutional monarchists, and so on).
        - **singular*** (String): Determines the singular adjective of the ideology type.
        - **description*** (String): Sets the description for the overall government/ideology type.

        - **effect** (Effect Scope): Determines the effects of the government.

- **Laws**
    - **Reform Category** (**Note:** If neither default_reform nor default_<government>_reform is actually set, the default reform will be based off of the political_appeasement scope. Failing that, the default reform that is set will be entirely random.
        - **name*** (String): Sets the human-readable name of the reform category. If not specified, it defaults to the name of the JSON key.

- **available_at_start*** (Boolean): Determines whether the reform category is available from the politics screen at the start of the game.
- **default_reform*** (String): Sets the default reform for all government types that do not have a further specified default_government_reform once the reform is unlocked, and applies their respective modifiers, but not their political_appeasement.
- **default_<government>_reform*** (String): Sets the default reform for the specified government type.

- **Reform**
  - **name*** (String): Determines the human-readable name of the reform.
  - **order*** (Integer): Sets the numeric order in which the reform should appear or take effect within the larger reform category.
  - **description*** (String): The base description for what the reform accomplishes.

  - **political_appeasement*** (Appeasement Scope): Determines the appeal the reform has to various political parties. Defaults to neutral for all ideologies if undefined.
  - **effects*** (Effect Scope): The effects that the reform has when passed. The bot will automatically attempt to apply the inverse of this whenever possible when a different reform in the category is passed. Modifiers in here are also automatically added to the tooltip.

  - **hide_modifiers*** (Boolean): Determines whether the modifiers and other effects inside the effects scope should be kept hidden from the player when viewed.

- **Localisation [WIP]**
  (**Note:** Everything in Localisation sits between curly brackets [when needing to be resolved to a string] and uses JSON key formatting, e.g. 'FROM.name'. Outside of curly braces, they become variables. They can also be used as JSON keys/scopes, where they take on contextual meaning based on their condition).
  - **FROM** (FROM scope, passes the current local scope that was passed to the original function).
    - **.*** Accesses the JSON value of the user scope, so long as it is a valid string.
  - **GLOBAL** (GLOBAL scope, accesses the main.global object instead).
    - **.*** Accesses the JSON value of the user scope, so long as it is a valid string.
  - **LOCAL** (LOCAL scope, passes any parameters as a JSON object).

        - **.*** Accesses the JSON value of the user scope, so long as it is a valid string.
    - **TO** (TO scope, passes the immediate local scope of the country that is interacting with the effect scope).
        - **.*** Accesses the JSON value of the user scope, so long as it is a valid string.

    - **RESEARCHED_TECHNOLOGIES_STRING** (National Scope): Returns a list of strings of all technologies a user has researched. This variable has the need of being truncated.

- **Modifiers** (**Note:** This section of the API is likely to be reworked in the future, as it contains too few options to be flexible enough).
    - **Modifier**
        - **default_value*** (Number): Sets the default value of the modifier. Defaults to 1 if not specified.
        - **name*** (String): Sets the human-readable name of a given modifier.
        - **type** (String, 'percentage'/'integer'): Determines what type of value a given modifier holds. Used especially in the UI when displaying user information.

        - **maximum*** (Number): Determines the maximum threshold the modifier is allowed to have.
        - **minimum*** (Number): Determines the minimum threshold the modifier is allowed to have.
        - **skeleton_modifier*** (Boolean): Defaults to false. Marks whether a modifier is a skeleton modifier or not on user displays. Skeleton modifiers are always hidden from the player as they are useless.

        - **create_capacity_<name>*** (Subcapacity Scope): Creates a new subcapacity of the given modifier which can function as a tracker and be referenced in other commands within the API.
        - **create_modifier_<name>*** (Submodifier Scope): Creates a new submodifier of the given modifier. This name can also be referenced in other commands within the API.

- **Pops**
    - **Pop Type**
        - **name*** (String): Sets the human-readable name of the pop type.
        - **singular*** (String): Sets the human-readable singular for the pop type.
        - **icon*** (Emoji String): Sets the UI icon for this pop type.

        - **chance*** (Percentage): Determines the proportion of the pop that typically spawns as this percentage. Note that this subtracts from the overall

‘generative pool’ of pops created upon generating a new colonised province object.
- **max_modifier_limit*** (Maximum Modifier Scope): Determines the upper threshold of modifiers gained from this pop type.
- **per_100k*** (Pop Effect Scope): Creates static modifiers per 100k of the population, with their effects being applied once per turn, but not to the modifiers themselves.
- **specialised_pop*** (Boolean): Determines whether the pop is specialised or not. Specialised pops have modifiers attached to them and can grant specific bonuses.

- **Technologies**
  - **Technology Category**
    - **Technology**
      - **name*** (String): Sets the human-readable name of the technology.
      - **icon*** (Emoji String): Sets the UI icon for this technology.
      - **description*** (String): The description for this technology.
      - **prerequisite_techs*** (String/Array): Techs that must be researched first before researching this one. If none are specified, the tech will be available for research at game start.
      - **research_cost** (Integer): How much Knowledge the technology should cost to research. (**Note:** If multiple research goods are present in the config, they are all able to count towards the same research_cost. This part of the API will likely be reworked at a future date).
      - **year*** (Integer): Used for calculating ahead-of-time buffs/debuffs. Defaults to 1500 if not specified.

      - **aliases*** (String/Array): Other names that this tech can be called by.
      - **custom_effect_description*** (String): Overrides all auto-generated effect descriptions.
      - **limit*** (Object, Limit Scope): Determines whether or not the tech is researchable by the player.
      - **mutually_exclusive*** (String/Array): Tech cannot be researched if any other technologies defined here have already been unlocked. Defaults to none.
      - **starts_researched*** (Boolean): Whether or not the tech should be automatically researched at game start.

      - **unlocks*** (Object, Effect Scope): What effects the technology should have upon completed research.
- **Units**
  - **Unit Category**

- **icon:** (Emoji String): Sets the UI icon for this unit category.
- **name:** (String): Sets the human-readable name of the unit category.
- **order:** (Integer): Determines how the unit category is ordered in comparison to other unit categories.
- **type*:** (String<Air, Naval, Land, Reserves>): Determines the type of the unit category.

- **Unit**
  - **name*** (String): Sets the human-readable name of the unit.
  - **icon*** (Emoji String): Sets the UI icon for this unit (defaults to army category icon if not defined).
  - **singular*** (String): Determines the human-readable singular name of the unit.
  - **type*** (String/Array): Separates the unit into various subcategories/tags for future scripting reference.

  - **attack*** (Integer): Sets the attack/damage value of the unit. Defaults to 0 if not defined.
  - **carrier_capacity*** (Integer): Determines the amount of aeroplanes an aircraft carrier can carry. This automatically means the unit becomes an aircraft carrier. Defaults to false.
  - **defence*** (Integer): Sets the hitpoint value of the unit. Defaults to 0 if not defined.
  - **manoeuvre*** (Integer): Sets the manoeuvre value of the unit (chance of dodging a hit). Defaults to 0 if not defined.
  - **initiative*** (Integer): Sets the initiative value of the unit (chance of attacking per battle tick). Defaults to 0 if not defined.

  - **colonise_provinces** (Integer): Determines how many provinces a given unit can colonise. Defaults to 0 if not defined.
  - **cost*** (Good List Scope): Determines the base cost for training the unit.
  - **manpower_cost*** (Manpower Scope): Determines the pop cost and effects of training the unit.
  - **quantity*** (Integer): Sets the quantity of the unit when crafted once. Defaults to 1.

- **Unit Types [WIP]**

List of Conditions:

- **any_country**: Resolves to true for any country that meets the specified subconditions. Switches to National Scope.
  - ***** (Conditions/Scopes)

- **good_required_per_population** (If Scope, Shortage - Number): Determines how much of the good per population is necessary to fulfil the limit.
- **has_flag** (If Scope, Object): Checks if the given country/scope has a flag set. Resolves to true if found, and to false if not.
    - **name** (String): The name of the flag to check for.
- **has_researched_technology** (If Scope, String): Returns true if a technology has already been researched by the target user, and false if it hasn't.
- **has_string** (If Scope, String/Object): Returns true if the target scope contains the following string.
    - **target** (String): Either a resolvable string or some other key. Fetches the current target and resolves it.
    - **value** (String): Checks if the current target contains the substring of this value. Returns true if found.
- **is_equal_to** (If Scope, String/Number/Object): Returns true if the target scope contains the following value.
    - **target** (String/Number): Either a resolvable string or number of some key. Can also be inputted as a raw value. Fetches the current target and attempts to resolve it.
    - **value** (String/Number): Checks if the current target is precisely equal to this value. Returns true if yes.
- **is_less_than** (If Scope, Number/Object): Checks whether a target scope is less than a certain number.
    - **target** (Number): Typically a key/variable that resolves to a number.
    - **value** (Number): Checks if the current target is less than this number. Returns true if yes.
- **is_less_than_or_equal_to** (If Scope, Number/Object): Checks whether a target scope is less than or equal to a certain number.
    - **target** (Number): Typically a key/variable that resolves to a number.
    - **value** (Number): Checks if the current target is less than or equal to this number. Returns true if yes.
- **is_greater_than** (If Scope, Number/Object): Checks whether a target scope is greater than a certain number.
    - **target** (Number): Typically a key/variable that resolves to a number.
    - **value** (Number): Checks if the current target is greater than this number. Returns true if yes.
- **is_greater_than_or_equal_to** (If Scope, Number/Object): Checks whether a target scope is greater than or equal to a certain number.
    - **target** (Number): Typically a key/variable that resolves to a number.
    - **value** (Number): Checks if the current target is greater or equal to this number. Returns true if yes.
- **random_chance** (If Scope, Number): Ran once each check. Resolves to true if the rolled chance lands within the specified percentage, resolves to false otherwise.
- **researched_technologies_is_greater_than** (If Scope, Number): Resolves to true if the number of researched technologies is higher than the given amount.

- **researched_technologies_is_less_than** (If Scope, Number): Resolves to true if the number of researched technologies is less than the given amount.
- **year** (If Scope, Number): Checks if the year is exactly a certain number.
- **year_is_less_than** (If Scope, Number): Checks if the year is less than a certain amount.
- **year_is_greater_than** (If Scope, Number): Checks if the year is greater than a certain amount.

List of Effects/Modifiers:

- **actions** (Integer): Adds or removes a set value of actions from a given nation.
- **add_cb** (Object, National Scope): Grants a CB to the target nation of the local scope.
    - **target** (String): Who to add the CB on to.
    - **type** (String): The type of CB to grant.
    - **duration*** (Integer): How long the CB should be valid for. Unlimited if unspecified.
- **add_expiry_effect** (Object, Global/National/Local Scope): Adds a constant effect to the local scope until conditions in limit return false.
    - **limit** (Limit Scope): All other Conditions/Effects/Modifiers in scope are instantly unapplied if conditions in limit return false.
    - **on_expiry*** (Object, Effect Scope): Applies everything in this scope recursively upon expiry of effect.
    - * (Conditions/Effects/Modifiers/Global Scope).
- **add_party_popularity** (Object, National Scope): Changes the popularity of a given party at the national level.
    - **type** (String): Changes the target of the party.
    - **value** (Percentage): Modifies the popularity of the selected party type by a given amount at the national level.
- **add_temporary_modifier** (Object, Global/National/Local Scope): Adds a temporary modifier to the given scope until the duration of it runs out.
    - **type** (String): The modifier to target.
    - **value** (Number): What to modify the local modifier by for the duration of the temporary modifier.
    - **duration** (Integer): How many turns the modifier should last for before being removed.

    - **name*** (String): An optional name that can be referred back to later on when removing temporary modifiers, or adjusting them.
    - **description*** (String): An optional description for the local modifier.
    - **effect*** (Object, Effect Scope): Attempts to reset and reapply everything in this scope every turn until the duration expires.
        - * (Conditions/Effects/Modifiers/Global Scope).
    - **on_expiry*** (Object, Effect Scope): Applies everything in this scope recursively upon modifier expiry.
        - * (Conditions/Effects/Modifiers/Global Scope).
- **add_to_variable** (Object, National/Local/Value Scope): Adds to a custom variable.

    - **name** (String): The variable name to add to.
    - **value** (Number/String/Object, becomes Value Scope): The value to add by.
- **add_unit** (Object, National Scope): Adds a certain amount of a unit to a nation's reserves.
    - **type** (String): The unit type to add.
    - **value** (Number): The value to add/deduct by.

    - **takes_manpower** (Boolean): Determines whether manpower is automatically deducted from a nation's pops or not.
- **add_vassal** (Object, National Scope): Vassalises a country.
    - **target** (String): Who to vassalise.
    - **overlord*** (String): Who should be the new overlord of the country. Defaults to local scope if not provided.
- **administrative_efficiency** (Percentage, National Scope): Adds/subtracts a given percentage from the administrative_efficiency modifier at the national scope.
- **<army_category>_ap/<army_category>_attack** (Integer/Percentage, National Scope): Increases/decreases the total attack damage of a specific category.
- **<army_category>_dp/<army_category>_defence** (Integer/Percentage, National Scope): Increases/decreases the total defence value of a specific army category.
- **army_professionalism** (Percentage, National Scope): Modifies the amount of professionalism present in a nation's military at the national scope.
- **army_travel_speed** (Percentage, National Scope): Modifies the rate at which armies travel for a given national scope.
- **army_upkeep** (Percentage, National Scope): Adds/subtracts a given percentage from the army upkeep for the nation at the national scope.
- **<building>** (Integer, National/Local Scope): Increases the individual building capacity for a single building.
- **<building_category>** (Integer, National Scope): Increases/decreases the overall building slot cap for a specific building category.
- **building_cost** (Percentage, National Scope): Modifies the base building cost for all buildings by a given amount.
- **can_justify_wars** (Boolean, National Scope): Determines whether the target can justify wars on foreign nations.
- **civilian_actions** (Percentage, National Scope): Increases/decreases the amount of civilian actions consumed by a country's population. Set at zero by default.
- **colonial_maintenance** (Percentage, National Scope): Modifies the amount of colonial_maintenance needed to maintain overseas colonies at the national scope by a given percentage.
- **colonial_power** (Percentage, National Scope): Modifies the base multiplier of Colonial Power available to a country by a certain percentage.
- **colonial_power_gain** (Percentage, National Scope): Modifies the rate of Colonial Power Gain at the national scope by a given percentage.

- **colonist_travel_speed** (Percentage, National Scope): Modifies the rate of travel for colonial units when colonising. Please note define limits in combat_defines regarding maximum colonial travel speed.
- **construction_speed** (Percentage, National Scope): Modifies the base construction_speed modifier of the given National Scope.
- **decrease_population_by** (Integer/Percentage, National Scope): Decreases the overall population of a country, target pops and provinces are selected at random.
- **decrease_relations** (Object, National Scope): Decreases relations with another country by a certain amount. Tied to local scope.
    - **target** (String): The target nation to decrease relations with.
    - **value** (Integer): How much relations should be decreased by.
- **diplomatic_slots** (Integer, National Scope): Increases/decreases the amount of diplomatic slots a nation is allowed to have.
- **disabled_reforms** (Boolean, National/Effect Scope): Determines whether reforms are disabled for the target nation/government.
- **divide_variable** (Object, National/Local/Value Scope): Divides a custom variable.
    - **name** (String): The variable name to divide by.
    - **value** (Number/String/Object, becomes Value Scope)**:** Resolves to either a number or a string; the value to divide the target variable by.
- **effect** (Object): Executes any scopes, conditions, or effects contained within the scope.
- **enable_blockades** (Boolean, National Scope): Enables/disables blockades for a given country.
- **enable_centralisation** (Boolean, National Scope): Enables the centralisation mechanic for this country.
- **enable_mobilisation** (Boolean, National Scope): Enables/disables mobilisation for a given country.
- **extra_building_slots** (Integer, National Scope): Increases/decreases the overall building slot cap for all building categories.
- **famine_penalty** (Percentage, National/Local Scope): Increases/decreases the impact that famines have on the nation/province. Starts at 0% by default unless overridden in governments file.
- **<good>** (Integer, National Scope): Adds/subtracts a certain amount of goods at the national level.
- **has_elections** (Boolean, National Scope): Determines whether the target has elections or not.
- **has_flag** (Object, Global/National/Local Scope): Returns either true or false depending on whether a flag is still active.
    - **name** (String): The name of the flag to check for.
- **<ideology>** (Percentage, National/Local Scope): Increases/decreases the ideology drift by a certain amount.
- **<ideology>_discontent** (Integer, National/Political Appeasement Scope): Raises/decreases the overall discontent for a given party. Usually contained within the political_appeasement scope.

- **improve_relations** (Object, National Scope): Improves relations with another country by a certain amount. Tied to local scope.
    - **target** (String): The target nation to improve relations with.
    - **value** (Integer): How much relations should be improved by.
- **increase_population_by** (Integer/Percentage, National Scope): Increases the overall population of a country, pops generated at random.
- **infamy_loss** (Percentage, National Scope): Modifies the infamy tickdown rate at the national level by a certain percentage.
- **inherit/annex_country** (Object, National Scope): Annexes an entire country into the ownership of another one.
    - **target** (String): The country to annex.
    - **overlord*** (String): The new country that should inherit the old one. Defaults to local scope if not specified.
- **join_war** (Object, National Scope): Joins a war on a certain side.
    - **target** (String): The ID of the war to join.
    - **side** (String, ‘defenders/attackers’): Determines which side of the war to join.
- **liberate_country** (Boolean, National Scope): Liberates a country from their overlord.
- **maximum_expeditions** (Integer, National Scope): Increases/decreases the number of maximum possible expeditions by the country.
- **maximum_manpower** (Percentage/National Scope): Increases/decreases the maximum manpower percentage from military pop types for a given nation.
- **maximum_tax_rate** (Percentage/National Scope): Determines the maximum tax rate for a given country, or what they can set their tax to.
- **<modifier>** (Percentage/Integer, National/Local Scope): Increases/decreases a modifier by a certain percentage or integer.
- **mobilisation_impact** (Percentage/National Scope): Increases/decreases the base multiplier for the impact of mobilisation on all modifier effects present in the base_mobilisation_imapct scope in combat defines.
- **mobilisation_size** (Percentage/National Scope): Increases/decreases the base multiplier for the amount of non-special pops available for mobilisation by a given nation.
- **mobilisation_speed** (Percentage/National Scope): Increases/decreases the base multiplier affecting the speed of mobilisation for the target national scope.
- **money** (Number): Adds/removes a certain amount of money to a nation.
- **multiply_variable** (Object, National/Local/Value Scope): Multiplies a custom variable.
    - **name** (String): The variable name to multiply by.
    - **value** (Number/String/Object, becomes Value Scope)**:** Resolves to either a number or a string; the value to multiply the variable by.
- **national_manpower** (Percentage, National Scope): Modifies the base multiplier of soldiers available for recruitment at any one time.
- **non_core_manpower** (Percentage, National Scope): Modifies the base multiplier of non-core manpower (defined as provinces with non-accepted cultures) available to the target country.
- **obsolete_building** (String/Array, National Scope): Obsoletes a list of buildings for the target scope.

- **obsolete_unit** (String/Array, National Scope): Obsoletes a list of units for the target scope.
- **opinion_modifier** (Integer, National Scope): Affects the opinion modifier of the target nation globally for all other countries.
- **overextension** (Percentage, National Scope): Modifies the base overextension modifier of the target nation globally by a given value.
- **political_capital** (Integer, National Scope): Adds/subtracts a certain amount of Political Capital from the national scope.
- **political_capital_gain** (Integer, National Scope): Increases the amount of Political Capital gained each turn.
- **pop_growth_modifier** (Percentage, National/Local Scope): Modifies the pop_growth_modifier of the given national scope. This is applied to all provinces, unless explicitly contained within a local province scope.
- **<pop_type>_growth_modifier** (Percentage, National/Local Scope): Modifies the pop_growth_modifier of an individual pop within the given national scope. Note that this does not override the global pop_growth_modifier, but rather multiplies to it. This is applied to all provinces as well, unless explicitly contained within a local province scope.
- **prestige** (Integer, National Scope): Adds/subtracts a given amount of prestige to the target National Scope.
- **production_efficiency** (Percentage, National Scope): Modifies the base production_efficiency modifier for all production buildings (defined as buildings that take in a certain amount of goods in exchange for an output of a certain amount of goods).
- **reform_desire_gain** (Percentage, National Scope): Increases/decreases global reform desire gain by a set amount.
- **republicanism** (Percentage, National Scope): Increases/decreases the amount of republicanism experienced by the target scope.
- **research_efficiency** (Percentage, National Scope): Increases/decreases the multiplier for the gain of all research goods within the country.
- **rgo_throughput** (Percentage, National Scope): Increases/decreases the base multiplier for raw resource gathering operations.
- **ruling_party_support** (Percentage, National/Local Scope): Increases the amount of support for the ruling party.
- **send_alert** (Object, National Scope): Sends an alert to a certain country, or player. These are similar to events, but only used for communicating.
    - **target** (String): The target country to send the alert to.
    - **type** (String): The type of alert to send. These are sent out from the alerts folder.
- **set_flag** (Object, Global/National/Local Scope): Adds a flag to the scope for either an indefinite or timed duration. Especially useful for cooldowns.
    - **name** (String): The name to give to the flag. This is so that it can be referred back to later.

    - **duration*** (Integer): Specifies how many turns the flag should last before being removed.

        - **effect*** (Effect Scope): Attempts to apply and reset all effects in this scope once per turn.
            - ***** (Conditions/Effects/Modifiers/Global Scope).
- **set_mobilisation_unit** (String): Sets the default mobilisation unit for a country.
- **set_party_popularity** (Object, National Scope): Sets the popularity for a given party, in contrast to **add_party_popularity**.
    - **type** (String): Changes the target of the party.
    - **value** (Percentage): The percentage of party popularity to change by. Please do not confuse this with an integer.
- **set_used_diplomatic_slots** (Integer, National Scope): Sets the amount of used diplomatic slots.
- **set_variable** (Object, National/Local Scope): Sets a target variable with a certain name and value.
    - **name** (String): Defines the name of the variable. If the variable name already exists, its existing value is overridden.
    - **type** (String - 'global', 'national', 'local'): Determines the scope of the current variable whenever applicable. 'global' applies the variable globally, 'national', applies the variable nationally, and 'local' automatically uses the current scope of the object.
    - **value** (Number/String/Object, becomes Value Scope): Resolves to either a number or string.
- **shipment_time** (Percentage, National Scope): Increases/decreases the multiplier for shipment_time at the given national scope. Please note the context of each modifier.
- **stability_modiifer** (Percentage, National Scope): Increases/decreases the overall National Stability of a country.
- **subtract_from_variable** (Object, National/Local/Value Scope) Subtracts from a custom variable.
    - **name** (String): The name of the variable to subtract from.
    - **value:** (Number/String/Object, becomes Value Scope): Resolves to either a number or string; the value to subtract from.
- **tax_efficiency** (Percentage, National Scope): Adds/subtracts a certain percentage from a nation's base tax_efficiency. Cannot exceed set maximum/minimum bounds present in the modifiers section of the config.
- **training_cost** (Percentage, National Scope): Adds/subtracts a certain percentage from the training cost of a given unit. Not to be confused with unit_cost, which modifies the base resource cost of a unit as opposed to the amount of Crafting Points they require.
- **<unit>** (Integer): Adds/removes a certain amount of a unit to a nation's reserves. Does not take up manpower, see **add_unit** for more detailed documentation.
- **unit_cost** (Percentage, National Scope): Adds/subtracts a certain percentage from the resource training cost of a given unit. Not to be confused with training_cost, which modifies the base CP cost of a unit as opposed to the amount of resources they require.
- **unlock_building** (String/Array, National Scope): Unlocks a building, or a list of buildings.
- **unlock_government** (String/Array, National Scope): Unlocks an available government/ideology.

- **unlock_reform** (String/Array, National Scope): Unlocks a series of reforms/reform categories.
- **unlock_unit** (String/Array, National Scope): Unlocks a list of units for the target scope.
- **use_diplomatic_slots** (Integer, National Scope): Takes/frees up a set number of diplomatic slots. Can exceed the diplomatic limit, so be careful.
- **war_exhaustion_rate** (Percentage, National Scope): Modifies the base war_exhaustion_rate applied each turn when at war. Note that this modifier is not applied during peacetime, including blockaded countries at peace.

List of Scopes:

Defines:

- **and/AND:** (And scope)
- **effect:** (Effect scope)
- **if/IF:** (If scope)
- **or/OR:** (Or scope)
- **not/NOT:** (Not scope)

Global Scopes:

- **And:** Returns true only if all of its constituent conditionals return true.
    - ***** (Conditions/Effects/Modifiers/Global Scopes)
- **Effect**
- **If**
    - **limit** (Limit Scope): All other Conditions/Effects/Modifiers in scope block only execute if all conditions in limit are true.
    - ***** (Conditions/Effects/Modifiers/Global Scopes)

    - **else***: Is checked whenever the limit is failed.
        - * (Conditions/Effects/Modifiers/Global Scopes)
- **Or**: Returns true if any of its constituent conditionals return true.
    - * (Conditions/Effects/Modifiers/Global Scopes)
- **Not**: Returns true only if none of its constituent conditionals return true.
    - * (Conditions/Effects/Modifiers/Global Scopes)

(**Note:** The following 'all scopes' can use the following commands: create_modifier_<name>, create_capacity_<name>).

- **All Building Categories**: Iterates over all specified building categories that obey the limit, similar to a for loop.
    - **limit*** (Limit Scope): All other Conditions/Effects/Modifiers in scope block only execute if all conditions in limit are true.
    - ***** (Conditions/Effects/Modifiers/Global Scopes)
- **All Good Types**: Iterates over all specified good types that obey the limit, similar to a for loop.
    - **limit*** (Limit Scope): All other Conditions/Effects/Modifiers in scope block only execute if all conditions in limit are true.

    - ***** (Conditions/Effects/Modifiers/Global Scopes)
- **All Pop Types**: Iterates over all specified pop types that obey the limit, similar to a for loop.
    - **limit*** (Limit Scope): All other Conditions/Effects/Modifiers in scope block only execute if all conditions in limit are true.
    - ***** (Conditions/Effects/Modifiers/Global Scopes)
- **All Units**: Iterates over all specified units that obey the limit, similar to a for loop.
    - **limit*** (Limit Scope): All other Conditions/Effects/Modifiers in scope block only execute if all conditions in limit are true.
    - ***** (Conditions/Effects/Modifiers/Global Scopes)
- **All Countries**: Affects all countries that meet the specified limit, similar to a for loop.
    - **limit*** (Limit Scope): All other Conditions/Effects/Modifiers in scope block only execute if all conditions in limit are true.
    - ***** (Conditions/Effects/Modifiers/Global Scopes)

Other Scopes:

- **Appeasement Scope**
    - **<government>_discontent*** (Integer): Determines the amount of discontent received by each government/ideology type when the scope is parsed.
- **Good List**
    - **<good>*** (Integer): Determines the amount of a certain type of good contained within the list.
- **Manpower Scope**
- **Max Modifier Scope**
    - **<modifier>*** (Number): Determines the upper bound of all affected static modifiers for the target scope (typically a Pop Scope).
- **Per 100k** (**Note:** Modifiers in this scope are typically kept static and do not affect the actual base multipliers).
    - ***** (Conditions/Effects/Modifiers/Global Scopes)
- **Pop Demand**
    - **required_by*** (String - ‘all’; Array): Determines the pop scope(s) the good is required by. ‘all’ sets the good to apply to all pops.
    - **type*** (String - ‘basic_need’): Determines the type of pop demand from the good.

    - **if_*** (If Scope, Shortage): Applies effects to population based on shortages, excesses, or other conditionals. Checked on a per turn basis.
- **Subcapacity Scope**
    - **name*** (String): Determines the human-readable name of the sub-capacity.
    - **aliases*** (String, Array): Sets any aliases that may be used in other config commands when referring to the capacity.

    - **default_value*** (Number): Sets the default value of the subcapacity when first initialised for the given scope.
    - **maximum*** (Number): Sets the maximum threshold of the subcapacity.

    - **minimum*** (Number): Sets the minimum threshold of the subcapacity.
- **Submodifier Scope** (**Note:** As opposed to subcapacities, submodifiers always take on the name of their parent as a prefix unless manually overridden).
    - **name*** (String): Determines the human-readable name of the sub-capacity.
    - **aliases*** (String, Array): Sets any aliases that may be used in other config commands when referring to the submodifier.
    - **negative*** (Boolean): Determines whether the given submodifier is a negative modifier or not. (Negative modifiers subtract instead of add, and have inverted operations).

    - **default_value*** (Number): Sets the default value of the submodifier when first initialised for the given scope.
    - **maximum*** (Number): Sets the maximum threshold of the submodifier.
    - **minimum*** (Number): Sets the minimum threshold of the submodifier.
- **Value Scope**
    - **set_value** (Number/String, Value Scope): Sets the initial value of the current value scope.

List of Variables:

Example Variable String: LOCAL_POPULATION * ((ATTACKER_AP_ROLL / DEFENDER_DP) / BUILDING_AMOUNT)

- **VARIABLE** (Local/National/Global Scope):

- **BUILDING_CATEGORY** (Building Category Scope): Fetches the human-readable name of the current building category scope.
- **FAMINE_IMPACT** (National Scope): Calculated level of impact from a famine.
- **GOOD_NAME** (Good Scope): Fetches the human-readable name of the current good scope.
- **<GOOD>_AMOUNT** (National Scope): Returns the total current amount of a good a user has.
- **<GOOD>_BUY_PRICE** (Global Scope): Fetches the global buy price of a good.
- **<GOOD>_NAME** (Global Scope): Returns the human-readable name of a specified good.
- **<GOOD>_SELL_PRICE** (Global Scope): Fetches the global sell price of a good.
- **<GOOD>_STOCK** (Global Scope): Fetches the current stock for a type of good in the World Market.
- **MONEY** (National Scope): Returns the total current amount of money a user has.
- **POP_NAME** (Pop Scope): Fetches the human-readable name of the current pop scope.
- **RANDOM_<MIN>,<MAX>** (Local/National/Global Scope): Returns a random number between MIN and MAX.
- **RESEARCHED_TECHNOLOGIES** (National Scope): Returns the total number of technologies a user has researched.

- **TARGET_COUNTRY** (Effect Scope): Returns the name of the target country.
- **UNIT_NAME** (Unit Scope): Returns the human-readable name of the current unit scope.