

Constantes, préférences et conditions By Module

Source serveur

`_utils/by_module/byModule_utils.js` construit un cache de constantes utilisé par l'ensemble du moteur. Les strings définies ici doivent correspondre aux clés des JSON de configuration.

Module preferences

Les préférences actuellement reconnues incluent :

```
num_decimals
our_company_type
company_legal_form
stock_type
has_taxes
has_deductions
has_decimals
has_formulas
uses_auxiliary_journals
has_employees
has_credit_notes
hidden_accounts
accounts_uses_last_financial_statement
uses_accounting_closing
unique_client_preference
unique_provider_preference
last_year_partners_with_empty_balance
uses_bill_id_in_general_journal
```

uses_new_regularizations
has_new_immobilizations
has_new_contracts
uses_regularizations
has_amortizations
regularization_accounts_count
pages
affected_regularizations_reports
has_immobilizations
has_immobilizations_taxes_report
uses_taxes_payment_form
uses_stock_regularizations
rate_num_bills_with_deductions
has_banking_transactions
has_land_immobilizations
manual_income_taxes
has_net_profit
permanent_stock_method
receive_or_send_taxes
decimal_money_tolerance_range
uses_bank_reconciliation
bank_reconciliation_form_numbers
module_time_limit
single_owner
partnership
joint_stock_a
joint_stock_ab
retry_summative
has_debts
has_interests
module_uid
ignore_grading_underline
ignore_grading_cash
ignore_grading_indent
ignore_grading_bold
auto_report_to_ledger

Page preferences

- `ignored_options;`
- `ignored_grading_column_uids;`
- `ignored_grading_relation_uids;`
- `money_tolerance_range;`
- `num_columns;`
- `has_dynamic_section.`

UIDs par défaut

Le framework utilise `default` comme UID standard pour table, section et row lorsque la structure ne nécessite pas d'identifiant spécifique.

Conditions de visibilité

Les conditions reconnues comprennent `condition_equal`, `or` et `and`, avec le préfixe `not_` pour l'inversion. Côté Unity, les classes `VisibilityCondition*` représentent ces conditions.

Résolution d'une préférence

Ne pas lire un JSON arbitrairement depuis un calcul. Le moteur passe par les helpers `getModulePreference()`, `getPagePreference()` et les settings du contexte courant, ce qui permet de gérer valeurs par défaut et overrides.

Où définir la valeur

Les commentaires du code indiquent la source attendue : plusieurs defaults sont dans `constants.json`, les préférences de pages dans `pages_settings.json` et certaines valeurs globales d'activité dans `act_settings.json`. La page protégée **Configuration Accounting — guide des JSON** reste la référence pour le détail des fichiers.

Ajouter une nouvelle préférence By Module

1. Déterminer le scope

- valeur globale par défaut → `constants.json`;
- valeur par module → `modules_settings.json` / `definitions`;
- valeur par page → `pages_settings.json` / `definitions`;
- valeur activité/région → `act_settings` / `regions_settings` selon le besoin.

2. Ajouter la constante string côté serveur

Ajouter la clé dans la table `modulesPreferences` ou `pagesPreferences` de `byModule_utils.js`. La string doit correspondre exactement au JSON.

3. Définir une valeur par défaut

Une preference manquante ne doit pas produire un comportement aléatoire. Ajouter le default dans la source appropriée et vérifier `getModulePreference()` / `getPagePreference()`.

4. Consommer via les helpers

Ne pas lire directement le JSON statique dans un moteur métier si l'helper de preference gère déjà overrides/context. Cela permet de conserver les régions et settings spécifiques.

5. Si Unity en a besoin

Ajouter la constante/lecture côté client seulement si le frontend doit réellement changer son comportement. Éviter de dupliquer une preference purement serveur.

Ajouter une condition de visibilité

Les conditions utilisent notamment `condition_equal`, `and`, `or` et le préfixe `not_`. Si un nouveau type de condition est nécessaire, il faut mettre à jour le parser/évaluateur côté Unity et, si utilisé côté serveur, sa logique équivalente.

Test minimal d'une nouvelle préférence

1. default absent d'override;
2. override module;
3. override région si applicable;
4. valeur false/0 explicite — ne pas la confondre avec “absente”;
5. ancien module sans la nouvelle clé;
6. Acc1 + Acc2 si la preference est partagée.

Revision #2

Created 2026-09-18 19:26:54 UTC by Mathis

Updated 2026-09-18 20:12:26 UTC by Mathis