# Validace balíčků

Každ balíček je validován v rámci svého kontextu, který je tvořen údaji z názvu balíčku:

* oboje
  + packageName – název balíčku bez přípon
* vzdálený přístup:
  + IDProvozovatel
  + Model
  + VykazovanéObdobí
  + DruhHry
* denní herní výkaznictví a denní herní zápis
  + IDProvozovatel
  + Model
  + VykazovanéObdobí
  + DruhHry
  + HerníMísto

Validace balíčků je popsána sadou konfiguračních souborů ve formátu CSV, které jsou zachyceny v jednotlivých záložkách Excelu <structured-schema-description.xlsx>.

Mechanizmus stahování balíčků zajišťuje, že hodnoty ve jméně balíčku jsou vždy správné, protože ke hledání balíčku je využíváno ověřených hodnot pocházejících z povolení. Název datového balíčku (bez přípon) odpovídá následujícím regulárním výrazům:

* vzdálený přístup
  + IDProvozovatel: [0-9]{9}
  + Model: [MV]
  + Datum: 20[1-9][0-9](01|02|03|04|05|06|07|07|09|10|11|12)(3[01]|012][0-9])
  + Čas: (00|08|16)
  + Typ hry: [LKRBTZ]
  + Verze: (0[1-9]|[1-9][0-9])

/[0-9]{9}[MV]20[1-9][0-9](?:01|02|03|04|05|06|07|07|09|10|11|12)(?:3[01]|[012][0-9])(?:00|08|16)[LKRBTZ](?:0[1-9]|[1-9][0-9])/

* denní výkaznictví
  + IDProvozovatel: [0-9]{9}
  + Model: [MV]
  + Měsíc: 20[1-9][0-9](01|02|03|04|05|06|07|07|09|10|11|12)
  + Typ hry: [LKRBTZ]
  + Herní místo: [0-9a-zA-Z-\_.]+
  + Verze: (0[1-9]|[1-9][0-9])

/[0-9]{9}[MV]20[1-9][0-9](?:01|02|03|04|05|06|07|07|09|10|11|12)[LKRBTZ](?:0[1-9]|[1-9][0-9])/

## Validace elektronické pečeti balíčku

Pokud nedojde o ověření elektronického pečeti, končí zpracování některou z následujících chyb:

* ERR\_PKG\_SIG\_INVALID\_DER – pokud není možné přečíst ASN.1 DER struktury balíčku
* ERR\_PKG\_SIG\_INVALID\_SIGNATURE – pokud nesouhlasí kryptografické ověření podpisu
* ERR\_PKG\_SIG\_CERT\_UNTRUSTED – pokud není možné ověřit platnost certifikátu
* ERR\_PKG\_SIG\_CERT\_UNKNOWN – pokud certifikát není mezi registrovanými certifikáty operátora

Libovolnou chybou validacepodpisu končí zpracování balíčku.

## Dešifrování balíčku

Pokud selže dešifrování balíčku, je ohlášena některá z chyb:

* ERR\_PKG\_ENC\_INVALID\_DER - pokud není možné přečíst ASN.1 DER struktury balíčku
* ERR\_PKG\_ENC\_RECIPIENT – pokud se nepodaří nalézt vhodný certifikát příjemce
* ERR\_PKG\_ENC\_BAD\_DATA – pokud se nepodaří dešifrovat

Libovolnou chybou dešifrování končí zpracování balíčku.

## Rozbalení ZIP archivu

Pokud se nepodaří rozbalit ZIP soubor balíčku, je zpracování ukončeno chybou ERR\_PKG\_UNZIP

Chybou rozbalení je ukončeno zpracování balíčku.

## Validace přítomných souborů – záložka “Package\_files”

Podle záložky „Package\_files“ je posuzováno, které soubory mají být v balíčku přítomné. Pokud balík obsahuje jiné soubory, než ty, které jsou uvedeny v tabulce, bude označen za chybný a zpracování se ukončí chybovým kódem ERR\_PKG\_EXTRA\_FILES a výčtem nadbytečných souborů.

Chybou kontroly přítomných souborů je ukončeno zpracování balíčku.

## Validace povinných souborů – záložka „Game\_types\_and\_files“

Je provedena operace získání seznamu souborů pomocí postupu, který lze zapsat pomocí SQL takto:

SELECT fileName FROM Game\_type\_and\_files as config

WHERE kontext.DruhHry=config.gameType AND kontext.Model=config.model

Pokud je množina názvů souborů neprázdná, pak získané soubory musí být v balíčku přítomny. Pokud požadovaný soubor(y) chybí, je zpracování balíčku ukončeno chybou s kódem ERR\_PKG\_MISSING\_FILE s uvedením výčtu chybějících souborů.

Pokud je množina prázdná, pak není prováděna žádná kontrola přítomnosti.

## Validace obsahu souborů

Každý soubor je načítán v rámci kontextu souboru, který přebírá všechny atributy kontextu balíčku a přidává kontext.fileName. Soubor se načítá řádek po řádku na každý řádek jsou aplikovány následující kontroly:

1. řádek je ukončen dvojicí znaků CRLF (0x0d 0x0a) - jiné znaky neukončují řádek
2. maximální délka řádku nesmí přesáhnout konfigurační parametr MAX\_CSV\_LINE\_LENGTH, který má výchozí hodnotu 64000. Délka řádku se počítá v bajtech. ERR\_LINE\_TOO\_LONG
3. řádek je tvořen výhradně validními znaky UTF-8 – ERR\_LINE\_BAD\_UTF8
4. pokud jde o první řádek v souboru, musí jeho obsah odpovídat regulárnímu výrazu:
   1. řádek je podle pravidel uvedených ve vyhlášce možno rozdělit na jednotlivá pole – ERR\_LINE\_INVALID\_CSV
   2. výsledkem předchozího kroku je sada hodnot jednotlivých polí, sada je validována podle
   3. je provedeno rozdělení pomocí znaku „;”, ze kterého vzniknou 4 hodnoty – pokud se nerozdělí na 4 hodnoty, skončí validace řádku chybou ERR\_LINE\_SPLIT\_META.
   4. metadata získaná rozdělením v předchozím kroku se validují takto:
      1. první pole se shoduje se jménem balíčku bez přípon – pokud se neshoduje (case insensitive) s názvem balíčku, končí validace pole chybou ERR\_META\_FIELD \_BAD\_PACKAGE
      2. druhé pole se shoduje s názvem souboru (case insensitive), pokud ne, končí validace pole chybou ERR\_META\_FIELD\_BAD\_NAME
      3. třetí pole obsahuje datum a čas ve formě shodné s validací doménového typu T\_DATETIME
      4. čtvrté pole obsahuje textovou hodnotu „1.0“
5. pokud jde o druhý řádek souboru, je podle pravidel rozdělení CSV řádku rozdělen na sadu položek hlavičky
6. Ze záložky „Fields“ se provede získání popisu obsahu hlaviček souboru analogicky k
   1. SELECT FieldNameCZ FROM Fields WHERE fileName=fileName AND model =model
7. Seznam položek hlavičky se musí shodovat se seznamem získaným ze záložky Fields. Pokud nesouhlasí, je načítání celého souboru ukončeno chybou ERR\_LINE\_BAD\_HEADER
8. Pokud se podař získat hodnoty z hlavičky, jsou zapamatovány v rámci konfigurace mapovací funkce fieldIdxToFieldName()
9. Pokud jde o jiný, než 1. nebo 2. nebo druhý řádek, provádí se validace datového řádku

Validace souboru je ukončena v případě, kdy je detekován dostatečný počet chybných řádku (konfigurační položka CFG\_MAX\_LINE\_ERR) a validace souboru končí chybou ERR\_FILE\_TOO\_MANY\_ERRORS.

## Validace datového řádku

1. Řádek je podle pravidel čtení CSV rozdělen na jednotlivá pole
2. Pokud je počet polí různý od počtu položek hlavičky souboru, je zpracování řádku ukončeno chybou ERR\_LINE\_BAD\_FIELD\_COUNT
3. Pro každou hodnotu pole se pomocí funkce fieldIdxToFieldName() zjistí, jaký je název pole (fieldName).
4. Pro každé pole jsou pak jsou provedeny kontroly podle podle obsahu odpovídajícího řádku v záložce Fields. Řádek se vybírá pomocí ekvivalentu:

SELECT \* FROM Fields

WHERE kontext.model=Model AND kontext.fileName=fileName AND fieldName=fieldName

## Validace polí

1. FieldType – podle hodnoty pole je provedena základní validace datového typu – ERR\_FIELD\_TYPE (viz záložka Field types)
2. DomainType – podle hodnoty pole je provedena základní validace doménového datového typu – ERR\_FIELD\_DOMAIN\_TYPE (viz záložka Domain types)
   1. Pokude je doménový typ D\_KOD, pak je ověřována existence hodnoty položky v číselníku. Číselník je identifikován podle názvy pole FieldNameCz. POkud není hodnota pole v číselníku nalezena, pak je reportována chyba ERR\_FIELD\_CODEBOOK
3. Nullable – podle hodny pole je provedena kontrola, zda je či nené povolena prázdná hodnota – ERR\_FIELD\_NOT\_NULL
4. Presence – podle hodnoty pole je provedena kontrola ERR\_FIELD\_MANDATORY
   1. mnandatory – musí být přítomna neprázdná hodnota
   2. mandatoryIf – pokud je splněna podmínka ve sloupci Condition, musí být přítomna hodnota
5. Condition – podmínka, která se vyhodnotuje pro použití mandatoryIf
   1. refFieldName.isEmpty – splněna, když hodnota refFieldName je prázdná
   2. refFieldName.notIn – splněna, když hodnota refFieldName není obsažena ve výčtu uvedeném v závorce
   3. refFieldName.In – splněna, když hodnota refFieldName je obsažena ve výčtu uvedeném v závorce
6. MinLength – minimální délka řetězce hodnoty počítaná ve znacích – ERR\_FIELD\_TOO\_SHORT
7. MaxLength – maximální délka řetězce hodnoty počítaná ve znacích – ERR\_FIELD\_TOO\_LONG
8. RegExp – obsah pole musí odpovídat regulírnímu výrazu – ERR\_FIELD\_BAD\_FORMAT\_RE
9. Context – obsah posuzovaného pole musí splňovat omezení vzhledem ke kontextu balíku:
   1. equalsGameType – shodný s kontext.gameType – ERR\_FIELD\_CTX\_GAMETYPE
   2. equalsOperatorId – shodný s kontext.OperatorId – ERR\_FIELD\_CTX\_OPERTORID
   3. equalsPackageDate – shodný s kontext.PackageDate – ERR\_FIELD\_CTX\_PGKDATE
   4. startsWithOperatorId – prefix hodnoty musíé být shodný s kontext.OperatorId – ERR\_FIELD\_CTX\_OPERATORID\_PREFIX
   5. withinPackageTimespan – hodnota data a času musí spadat do časového rozsahu, který je pokryt balíčkem. ERR\_FIELD\_CTX\_PKG\_TIMESPAN

Pokud dojde k chybě validace některého z polí, je po provedení všech validací oznámena chyba ERR\_LINE\_INVALID\_FIELDS.

## Struktura chybových hlášení

Každá chyba začíná na novém řádku chybovým kódem některého z typů:

* /^ERR\_PKG\_.\*: /
* /^ERR\_FILE\_.\* [jmenosouboru.csv]: /
* /^ERR\_LINE\_.\* [jmenosouboru.csv][čísloŘádky]: /
* /^ERR\_FIELD\_.\* [jmenosouboru.csv][čísloŘádky][fieldName/fieldPosition]: /

Každý chybový záznam může být zapsán na více řádků.

Tok zpracování je následující:

* ERR\_PKG – ukončuje zpracování balíčku
* ERR\_FILE – ukončuje zpracování souboru
* ERR\_LINE – ukončuje zpracování řádky
* ERR\_FIELD – vždy jsou provedeny všechny dostupné validace na položce

## Struktura chybového reportu

Chybový report obsahuje vždy chyby podle popisu z předchozích kapitol tak, jak vznikaly při zpracování.

## Struktura OK reportu

Potvrzovací report obsahuje jedinou řádku:

OK: YYYYMMDD-HHMMSS

Kde datum a čas zachycuje okamžik generování OK reportu.