

MASARYKOVA UNIVERZITA
FAKULTA INFORMATIKY



Nástroj na úpravy hlaviček FITS súbtorov

BAKALÁRSKA PRÁCA

Martin Vrábel

Brno, jeseň 2015

Prehlásenie

Prehlasujem, že táto bakalárska práca je mojím pôvodným autorským dielom, ktoré som vypracoval samostatne. Všetky zdroje, pramene a literatúru, ktoré som pri vypracovaní používal alebo z nich čerpal, v práci riadne citujem s uvedením úplného odkazu na príslušný zdroj.

Martin Vrábel

Vedúci práce: RNDr. Martin Kuba, Ph.D.

Podakovanie

Podakovanie patrí vedúcemu práce, RNDr. Martinovi Kubovi, Ph.D., za užitočné rady a trpezlivosť, konzultantom, doc. RNDr. Miloslavovi Zejdovi, Ph.D. a Mgr. Marekovi Skarkovi, Ph.D., ktorí zároveň testovali vývojové verzie nástroja, za ich odbornú pomoc a tipy na zlepšenie. Ďalej chcem poďakovať rodine a kamarátom nielen za podporu pri vypracovávaní práce, ale aj počas celého štúdia.

Zhrnutie

Cieľom bakalárskej práce je vytvoriť nástroj na hromadnú úpravu hlavičiek FITS súborov pre Ústav teoretickej fyziky a astrofyziky Prírodovedeckej fakulty Masarykovej univerzity v Brne. Rozsahom práce je analýza požiadavkov, popis FITS formátu a použitých technológií, návrh hlavičkových operácií, implementácia výkonnej a grafickej časti aplikácie a ich podrobný popis.

Klíčové slová

FITS, Java, JavaFX, Header Data Unit, klíčové slovo, hlavička, komentár, hlavičkový záznam

Obsah

1	Úvod	1
2	Rozbor požiadaviek	3
2.1	<i>Doplnenie požiadaviek počas vývoja</i>	4
3	FITS formát	5
3.1	<i>História vývoja</i>	5
3.2	<i>Štruktúra</i>	6
3.2.1	Hlavička	7
3.2.2	Dátová časť	8
3.3	<i>Štruktúra hlavičky</i>	8
3.3.1	Kľúčové slovo	8
3.3.2	Hodnota	9
3.3.3	Komentár	10
3.4	<i>Príklady vybraných kľúčových slov</i>	11
4	Použité technológie	14
4.1	<i>Dostupné knižnice na prácu s FITS</i>	16
5	Implementácia	18
5.1	<i>Finálny návrh hlavičkových operácií</i>	18
5.2	<i>Výkonná časť</i>	19
5.2.1	Spracovanie vstupov	20
5.2.2	Validácia vstupných dát	22
5.2.3	Úprava hlavičiek zadaných súborov	23
5.2.4	Spracovanie výstupu z operácií	26
5.2.5	Ostatné časti	28
5.2.6	Spustenie programu	30
5.3	<i>Grafická časť</i>	30
5.3.1	Zoznam FITS súborov	33
5.3.2	Panel s operáciami	34
5.3.3	Textový výstup	37
5.3.4	Lokalizácia	38
5.3.5	Nastavenia programu	38
6	Súvisiace práce	40
7	Záver	42
	Literatúra	44
A	Prílohy	46

1 Úvod

Vedci a vzdelanci už v starovekom Grécku či Mezopotámii pozorovali oblohu a pohyb hviezd na nej. Tieto pozorovania si vtedy zaznamenávali a ručne zakresľovali do zvitkov a tabuliek. Ako sa astronómia počas nasledujúcich desiatok stoviek rokov vyvíjala až do dnešnej podoby, pozorovania oblohy sa stali čoraz zložitejšie a presnejšie a začalo sa z nich generovať stále viac a viac dát. Na ich uloženie bolo potrebné vymyslieť spoločný formát, ktorý by umožňoval snímky a dáta z pozorovaní efektívne skladovať a prenášať. Z tejto potreby vznikol v roku 1981 prvý štandard formátu **FITS – Flexible Image Transport System** [1].

Na čítanie a spracovanie súborov vo formáte FITS dnes existuje veľa rôznych nástrojov a programov, so základnou alebo rozšírenou funkcionalitou. Mnohé z nich dokážu upravovať hlavičku tohto súboru, avšak len minimum dokáže tieto operácie vykonávať nad viacerými súbormi súčasne.

Aj kvôli tomuto stavu vznikla požiadavka na vývoj nástroja, ktorý umožní upravovať hlavičky viacerých FITS súborov naraz. Práca sa venuje jeho návrhu a vývoju a výsledkom má byť program alebo sada programov, ktoré umožnia vykonávať takéto hromadné úpravy.

Program sa má skladať z dvoch častí: **výkonnej časti**, ktorá bude spustiteľná z príkazového riadka a bude vykonávať úpravy hlavičiek FITS súborov, a **grafickej časti**, ktorá umožní výber súborov a zadanie úprav pohodlne myšou a bude volať výkonnú časť na samotnú úpravu hlavičiek.

Jednou z požiadaviek je aj možnosť spustenia programu v systémoch MS Windows a Linux, preto je práca napísaná v objektovo orientovanom jazyku **Java**, ktorý túto požiadavku spĺňa. Grafické rozhranie navyše používa jeho nadstavbu – jazyk **JavaFX**, ktorý slúži na tvorbu moderných užívateľských rozhraní a aplikácií.

Textová časť práce sa okrem úvodu a záveru skladá z piatich kapitol.

V úvodnej kapitole sa nachádza súhrn požiadaviek od zadávateľa práce a ich podrobná analýza.

Ďalšia kapitola sa zaoberá stručnou históriou formátu FITS a detailnejšie popisuje jeho štruktúru, najmä štruktúru hlavičky a jej znakov.

V nasledujúcich kapitolách je už podrobný popis návrhu a implementácie nástroja, jednotlivých častí a procesov. Sú rozobraté aj technológie a externé knižnice použité pri vývoji. Tieto kapitoly tvoria najväčšiu časť práce.

V poslednej kapitole sa nachádza porovnanie už existujúcich riešení a programov, ktoré sa funkcionalitou približujú zadaniu.

V závere sú zhrnuté dosiahnuté výsledky práce a načrtnuté možnosti ďalšieho rozšírenia funkcionality programu či nadviazania na túto prácu.

V prílohách sa nachádzajú samotné programy spolu s ich zdrojovými kódmi, používateľský manuál s popisom použitia hlavičkových operácií a nezverejnený dokument odkazovaný v textovej časti práce.

Keďže sa jedná o zadávateľa z reálneho prostredia, je pravdepodobné nasadenie výsledných programov v praxi.

2 Rozbor požiadaviek

Cieľom tejto práce je vytvoriť nástroj pre Ústav teoretickej fyziky a astrofyziky Prírodovedeckej fakulty Masarykovej univerzity, ktorý umožní hromadnú manipuláciu so záznamami hlavičiek FITS súborov, ich úpravu, pridávanie, odoberanie a iné. Dôraz je kladený na hromadnosť úprav, aby bolo možné naraz upraviť záznamy viacerých súborov (napríklad z konkrétneho pozorovania) a nemuseli byť upravované jednotlivo.

Potreba pre vznik takéhoto nástroja vzišla z nejednotnosti medzi FITS súbormi produkovanými rôznymi CCD kamerami. Niektoré údaje o snímku chýbali, iné boli zapísané chybné alebo označené rôznymi kľúčovými slovami, aj keď sa jednalo o ten istý údaj (napr. EXPOSURE, EXPTIME, INT pre expozičnú dobu). Astronómovia chcú snímky ukladať do archívu, v ktorom sa má vyhľadávať podľa určitých kľúčových slov, preto chcú mať pred uložením možnosť zjednotiť a opraviť tieto údaje a dosiaľ nemali žiadny podobný nástroj.

Program sa má skladať z dvoch častí:

1. **výkonnej časti** spustiteľnej z príkazového riadku alebo skriptu, ktorá bude vykonávať operácie s hlavičkami
2. **grafickej časti** na pohodlný výber súborov a zadanie úprav záznamov

pričom grafická časť bude využívať tú výkonnú a volať ju ako oddelený proces.

Výkonná časť musí umožňovať základné operácie s hlavičkami FITS súborov:

- pridanie nového hlavičkového záznamu
- úpravu existujúceho záznamu
- odstránenie existujúceho záznamu
- posun časovej hodnoty existujúceho záznamu o zadanú hodnotu

- zrefazenie ľubovoľného počtu existujúcich záznamov a konštánt do jedného záznamu (nového alebo už existujúceho, ako jeho aktualizáciu)

Grafická časť musí spĺňať nasledovné požiadavky:

- výber adresára s FITS súbormi na úpravu
- filtráciu súborov v adresári podľa názvu s využitím tzv. *globbingu*
- filtráciu súborov podľa toho, či obsahujú záznam so zadaným kľúčovým slovom alebo hodnotou
- výber buď všetkých, alebo iba niektorých súborov pomocou myši

Zadanie bližšie nešpecifikuje použitý programovací jazyk, program však musí byť spustiteľný minimálne v operačných systémoch **MS Windows** a **Linux**.

2.1 Doplnenie požiadaviek počas vývoja

Počas vývoja programu a po konzultácii so zadávateľmi práce došlo k doplneniu požiadaviek na základnú funkcionality programu o ďalšie operácie s hlavičkovými záznamami:

- prepočet štandardného času a dátumu pozorovania do **Juliánskeho dátumu**
- výpočet **Heliocentrického Juliánskeho dátumu** z času a dátumu pozorovania a zadaných súradníc pozorovaného objektu

pričom sa vypočítané hodnoty uložia do hlavičky súboru ako nový záznam, alebo ako aktualizácia už existujúceho záznamu. V prípade Heliocentrického Juliánskeho dátumu je potrebné uložiť do hlavičky aj zadané súradnice pozorovaného objektu.

3 FITS formát

Flexible Image Transport System (FITS) je otvorený štandard, ktorý definuje formát dát používaný na ukladanie, prenos a spracovanie vedeckých a iných snímok [2]. Najviac rozšírený je v astronómii, kde slúži na uchovávanie pozorovateľských snímok z CCD kamier spolu s ďalšími informáciami a metadátami.

Pôvodne bol vytvorený na ukladanie snímok, ktoré pozostávali z dvoj a viacrozmerného bloku dát – samotného obrázku – a hlavičky, ktorá popisovala jeho organizáciu a obsah. Neskôr bol štandard rozšírený, aby dokázal zahrnúť aj komplexnejšie dáta a vyhovoval rôznym typom astronomických meraní, ako napríklad spektrometrii, polarimetrii a iným.

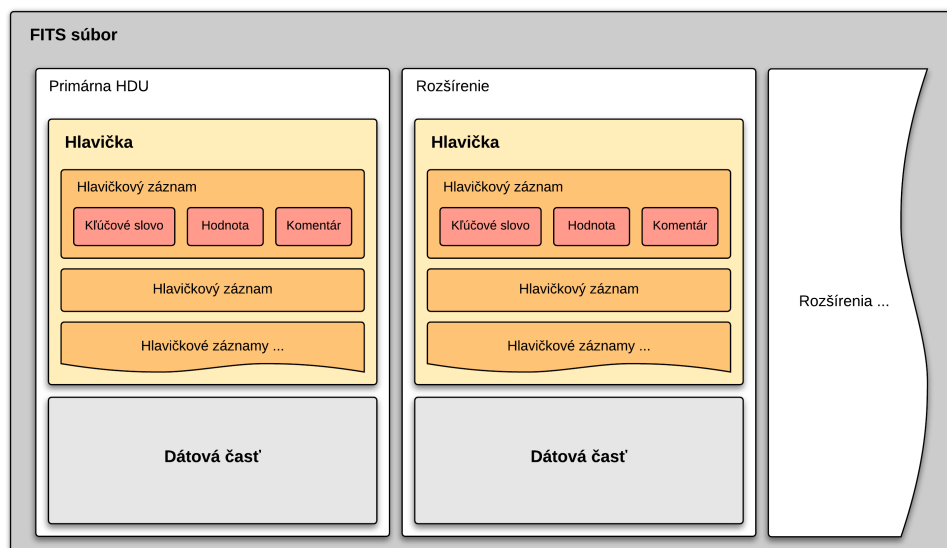
3.1 História vývoja

Počiatočná definícia a popis FITS formátu pochádza z práce „FITS : A Flexible Image Transport System“ [1] z roku 1981. O rok neskôr bol formát oficiálne schválený **Medzinárodnou astronomickou úniou** (IAU) a postupne bola pridaná podpora pre prvé rozširujúce štruktúry (ASCII tabuľky). V roku 1988 bola Medzinárodnou astronomickou úniou vytvorená pracovná skupina **IAU FITS Working Group**¹ (IAU-FWG), ktorá bola poverená spravovaním, udržiavaním a ďalším vývojom FITS formátu. V roku 1993 bol vydaný prvý oficiálny štandard **NOST 100-1.0** [3] oddelením NASA pre vedecké štandardy a technológie². V nasledujúcich verziách pribudla podpora pre ďalšie rozšírenia (napr. binárne tabuľky, 64-bitové číselné dátové typy a iné). Od verzie **2.1** z roku 2005 bola štandardizácia prevzatá pracovnou skupinou IAU-FWG, ktorá schválila a v roku 2010 vydala zatiaľ posledný a doteraz aktuálny štandard verzie **3.0** [4].

1. domovská stránka pracovnej skupiny: <http://fits.gsfc.nasa.gov/iaufwg/iaufwg.html>

2. **NOST** - NASA/Science Office of Standards and Technology

3.2 Štruktúra



Obr. 3.1: Vizualizácia štruktúry typického FITS súboru (podľa [5])

FITS súbor sa skladá z jednej alebo viacerých častí zvaných **HDU** – **H**header + **D**ata **U**nit. Prvá HDU časť sa nazýva **primárna HDU**. Môže byť buď prázdna, alebo obsahovať n-rozmerné pole dát – zvyčajne obrazovú informáciu (1D spektrum, 2D snímka atď.). Každá ďalšia časť sa volá **FITS rozšírenie**. Podľa štandardu [4] môžu byť rozšírenia trojakého typu:

- **obrazové** – rovnako ako pri primárnej HDU
- **ASCII tabuľka** – stĺpce a riadky dát v ASCII³ formáte
- **binárna tabuľka** – stĺpce a riadky dát reprezentované v binárnom formáte

FITS súbor, ktorý obsahuje iba primárnu HDU časť, sa nazýva základný FITS súbor alebo *Single Image FITS (SIF)*. Súbor, ktorý obsahuje

3. ASCII – American Standard Code for Information Interchange = kódovacia znaková sada, viac na <http://www.asciitable.com/>

jedno alebo viac rozšírení, sa nazýva *Multi-Extension FITS (MEF)*. Za validný sa považuje aj súbor, ktorý má primárnu HDU prázdnu a obsahuje iba FITS rozšírenia. [6]

Ako názov napovedá, každá HDU časť je zložená z **hlavičky** a **dátovej časti**.

3.2.1 Hlavička

Hlavička obsahuje metadáta uložené po riadkoch v – pre človeka čitateľnom – formáte v ASCII znakoch (povolené sú však iba tlačiteľné znaky s decimálnym kódom 32–126). Jeden riadok sa nazýva **hlavičkový záznam**, má nasledovný tvar:

kľúčové slovo = hodnota / komentár

a pevne stanovenú dĺžku maximálne **80** bajtov. Celá hlavička môže zaberať maximálne **2880** bajtov. V hlavičke tak môže byť najviac **36** záznamov / riadkov, pričom musí byť vždy ukončená záznamom s kľúčovým slovom END. V prípade menšieho počtu záznamov je zvyšok hlavičky doplnený medzerami (decimálny kód 32). Štruktúra hlavičky je bližšie popísaná v časti 3.3.

Záznamy sa podľa kľúčových slov rozdeľujú na 4 skupiny:

- **povinné** – musia byť uvedené v každej hlavičke a to práve raz, napr. SIMPLE, BITPIX a i.
- **povinné pre určitý typ snímky** – musia byť uvedené v hlavičke daného typu práve raz, pre ostatné typy sú rezervované, napr. FILTER – pre *Light Frame* a *Flat Field* snímky
- **rezervované** – ak sú dostupné aj spolu s hodnotami (záleží napríklad na type CCD kamery), uvedú sa v hlavičke v požadovanom formáte
- **ostatné** – nové a iné, ktoré splňajú predpísaný štandard

Záznamy s rezervovanými a ostatnými kľúčovými slovami by sa takisto nemali v hlavičke objaviť viac než raz.

3.2.2 Dátová časť

Dátová časť nasleduje ihneď po hlavičke. Nie je povinná, takže niektoré HDU časti ju nemusia obsahovať vôbec. Jej prítomnosť je indikovaná hodnotou hlavičkového záznamu s kľúčovým slovom NAXIS.

Primárna HDU časť a **obrazové rozšírenie** obsahujú dáta snímky, pričom jeden jej pixel môže byť uložený v niektorom z piatich podporovaných dátových typov:

- **8-bitový** celočíselný dátový typ bez znamienka
- **16-bitový** celočíselný dátový typ so znamienkom
- **32-bitový** celočíselný dátový typ so znamienkom
- **32-bitové** reálne číslo s plávajúcou desatinnou čiarkou
- **64-bitové** reálne číslo s plávajúcou desatinnou čiarkou s dvojitou presnosťou

Zvyšné dve rozšírenia – **ASCII a binárne tabuľky** – obsahujú informácie zoradené v *riadkoch* a *stĺpcoch*. Každý stĺpec môže mať samostatný dátový typ, pričom binárne tabuľky podporujú aj zložitejšie dátové typy. Práve binárne tabuľky sú všeobecne účinnejšie a komplexnejšie ako ASCII tabuľky. Čítanie a zápis dát prebieha rýchlejšie z dôvodu lepšej reprezentácie dát pre strojové spracovanie a každé pole tabuľky dokáže uchovať pole dát namiesto obyčajnej skalárnej hodnoty v ASCII tabuľkách [2][4].

3.3 Štruktúra hlavičky

3.3.1 Kľúčové slovo

Kľúčové slovo je 8-znakový, doľava zarovnaný ASCII reťazec. Tvorí prvú časť hlavičkového záznamu, v ktorom je umiestnený na pozícii 1–8. Môže obsahovať iba **čísllice** (decimálny kód 48–57), **veľké písmená** (65–90), znak **podčiarkovník** _ (95), **spojovník** - (45) a **medzeru** □ (32), tá sa však nemôže nachádzať uprostred reťazca, slúži iba na vyplnenie dĺžky v prípade, že je kratší ako 8 znakov.

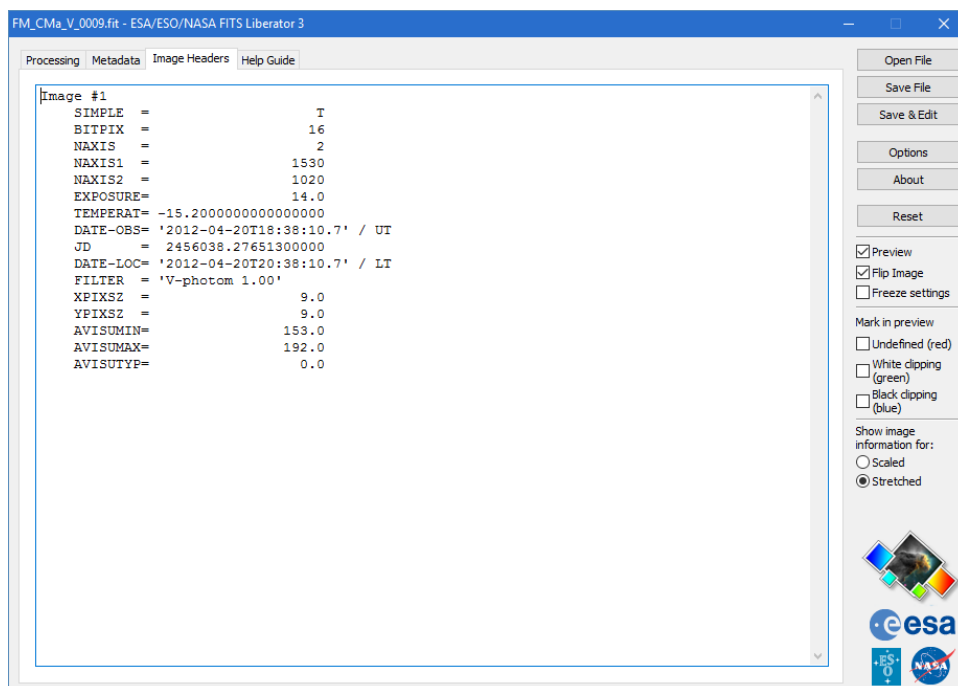
3.3.2 Hodnota

Ak sa na pozíciách 9 a 10 nachádzajú znaky *rovná sa* = a *medzera* , znamená to, že nasleduje **hodnota** záznamu. Existujú totiž kľúčové slová, ktoré z definície nemajú žiadnu hodnotu, napr. CONTINUE, COMMENT alebo prázdne kľúčové slovo.

Zvyšnú časť záznamu, teda pozície 11 až 80, zaberá hodnota, prípadne ju môže nasledovať komentár. Ak je ich celková dĺžka kratšia, záznam je do pozície 80 vyplnený opäť medzerami.

Hodnota môže nadobúdať rôzne dátové typy, podľa toho sa odvíja aj formát jej uloženia v zázname:

- **logický** dátový typ – hodnota T (*true*) alebo F (*false*) uložená na pozícii 30
- **celočíselný** dátový typ – číselná hodnota zarovnaná doprava na pozíciách 11–30, začína znamienkom + (ASCII 43 decimálne) alebo – (ASCII 45 decimálne), pričom + je nepovinné, vodiace nuly sú takisto nepovinné
- **reálne číslo** s plávajúcou desatinnou čiarkou – číselná hodnota je opäť zarovnaná doprava na pozíciách 11 až 30, začína znamienkom + (nepovinné) alebo –, nasleduje sekvencia číslic, ktorá musí obsahovať práve jednu desatinnú bodku . (ASCII 46 decimálne) oddeľujúcu celú a desatinnú časť čísla; prípadný *exponent* sa zapíše ako písmeno E (ASCII 69 decimálne) nasledované celým číslom – exponentom
- **textový reťazec** – môže zabrať všetky bajty na pozíciách 11–80, vždy musí byť ohraničený jednoduchými úvodzovkami ' (ASCII 39 decimálne), maximálna dĺžka je 68 znakov bez úvodzoviek; predchádzajúce verzie štandardu vyžadovali, aby bol kratší reťazec doplnený zľava medzerami do minimálnej dĺžky 8 znakov, takže uzatváracia úvodzovka bola vždy najskôr na pozícii 20 – v novej verzii to už nie je vyžadované, zapisuje sa tak iba kvôli spätnej kompatibilite

Obr. 3.2: Ukážka základnej FITS hlavičky v programe FITS Liberator⁵

3.3.3 Komentár

Dodatočné informácie o zázname je možné vložiť do komentára. Hneď po hodnote musí nasledovať **medzera** `␣` (nepovinná, ale veľmi odporúčaná) a po nej znaky **lomka** `/` a **medzera** `␣`, ktoré indikujú jeho začiatok. Komentár obsahuje textový reťazec z povolených ASCII znakov (32–126 decimálne).

Ak je hodnota záznamu číselná alebo logická, komentár začína na pozícii 34 – v tomto prípade je jeho maximálna dĺžka 47 znakov. Ak je hodnotou textový reťazec, komentár sa môže nachádzať aj skôr, hneď za reťazcom.

5. Prehliadač FITS súborov – dostupný z https://www.spacetelescope.org/projects/fits_liberator/

Komentár často slúži aj na zapisovanie jednotiek fyzikálnych veličín k daným hodnotám. Sú uvedené v hranatých zátvorkách hneď na začiatku, voliteľne nasledované ďalším popisom:

EXPTIME = 120.0 / [s] exposure time

3.4 Príklady vybraných kľúčových slov

SIMPLE

status: povinné
dátový typ: logický
popis: Vždy prvé kľúčové slovo primárnej hlavičky všetkých FITS súborov. Obsahuje hodnotu T ak súbor spĺňa štandard. V opačnom prípade obsahuje hodnotu F. Nesmie sa nachádzať v hlavičkách rozšírení.

BITPIX

status: povinné
dátový typ: celočíselný
popis: Hodnota špecifikuje spôsob uloženia dátovej časti súboru, konkrétne počet bitov. Musí byť uvedené hneď za kľúčovým slovom SIMPLE a hodnota môže nadobúdať jednu z povolených hodnôt: [8, 16, 32, 64, -32, -64].

NAXIS

status: povinné
dátový typ: celočíselný
popis: Obsahuje nezáporné celé číslo od 0 do 999, ktoré definuje počet osí v príslušnej dátovej časti. Ak sa rovná nule, za hlavičkou nenasledujú žiadne dáta. Musí byť uvedené hneď za kľúčovým slovom BITPIX

NAXISn

status: povinné (ak NAXIS > 0)
dátový typ: celočíselný
popis: Udáva dĺžku **n**-tej osi, teda počet prvkov na osi **n** v dátovej časti súboru. Napríklad pri 2D snímkach určuje rozlíšenie v pixeloch. Povinné sú všetky kľúčové slová od 1 až po hodnotu NAXIS vo vzostupnom poradí (NAXIS1, NAXIS2 ...) a žiadne ďalšie. Uvedú sa hneď za kľúčovým slovom NAXIS.

EXTEND

status: nepovinné
dátový typ: logický
popis: Kľúčové slovo sa môže objaviť iba v **primárnej hlavičke**, nesmie sa byť prítomné v hlavičke rozšírenia. Hodnota T označuje, že sa v súbore môžu nachádzať rozšírenia (ale aj nemusia, kľ. slovo je iba pomocné).

CONTINUE

status: rezervované
dátový typ: nedefinovaný
popis: Slúži na rozdelenie príliš dlhého textového reťazca predchádzajúceho záznamu do viacerých záznamov. Po kľúčovom slove nasleduje dvojica medzier na pozíciách 9 a 10 a po nich pokračovanie predchádzajúceho reťazca, opäť musí byť ale uzatvorený v jednoduchých úvodzovkách `'`. Na koniec predchádzajúceho záznamu sa pridá znak `&` (ASCII znak 38 decimálne), ktorý indikuje pokračovanie reťazca práve v ďalšom zázname s kľúčovým slovom CONTINUE.

Príklad:

```
VALUE      = 'This is a long string value&'
CONTINUE   = ' that spans&'
CONTINUE   = ' over three lines.'
```

Rozšírený popis hlavičky a ďalšie kľúčové slová sú popísané v prílohe 2 .

4 Použité technológie

Základným stavebným kameňom pre vývoj nástroja bol výber programovacieho jazyka. Ako hlavný programovací jazyk pre výkonnú aj grafickú časť bol vybraný jazyk **Java**. Spĺňa základné požiadavky na multiplatformnosť, teda program bude spustiteľný v operačných systémoch *MS Windows* a *Linux*, navyše je možné programy v ňom napísané spúšťať aj v systémoch *Apple Mac OS X* a to bez akýchkoľvek ďalších úprav. Podľa rôznych štatistík^{1,2} sa radí medzi najrozšírenejšie jazyky súčasnosti, je stále vyvíjaný a rozširiteľný vďaka množstvu voľne dostupných knižníc a nástrojov. Medzi ne patria aj knižnice na spracovanie FITS súborov, ktoré sú bližšie popísané v časti 4.1. Použitá je najnovšia a aktuálne podporovaná verzia **Java 8**.

Grafická časť programu je postavená na nadstavbe jazyka Java – **JavaFX** vo verzii **8**, ktorá sa používa na tvorbu moderných grafických rozhraní a aplikácií v jazyku Java. Má nahradiť dnes už zastarané rozhranie *Swing* používané ako štandard v starších verziách jazyka Java. Od aktuálnej verzie je *JavaFX* už súčasťou základnej distribúcie.

Na vývoj vizuálnych častí grafického rozhrania bol použitý nástroj **SceneBuilder**³. Je to vizualizačný nástroj pre balík *JavaFX* ktorý umožňuje vytvoriť grafickú stránku programu tzv. *WYSIWYG*⁴ spôsobom, teda priamym rozmiestnením elementov na obrazovke. Pôvodne vyvíjaný firmou *Oracle*, od verzie 2.2 ako voľne dostupný softvér, v jeho vývoji sa rozhodla pokračovať spoločnosť *Gluon* vydaním verzie **8.0**⁵ (podľa *JavaFX 8*).

Na správu externých knižníc a automatizovaného zostavovania a testovania programu bol použitý nástroj **Apache Maven 3** [7]. Jeho základom je **XML súbor**, v ktorom sa popíšu všetky závislosti projektu

1. TIOBE Index – mesačne aktualizovaný rebríček popularity programovacích jazykov, zostavovaný podľa rôznych kritérií, viac na <http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html>

2. GitHub – štvrťročne aktualizovaný prehľad vytvorený zo štatistík programovacích jazykov repozitárov na populárnom verzovacom serveri GitHub, dostupný na <http://github.info/>

3. Dostupný z <http://gluonhq.com/open-source/scene-builder/>

4. **What You See Is What You Get** - vo voľnom preklade: „dostaneš to, čo vidíš“, viac informácií na <https://en.wikipedia.org/wiki/WYSIWYG>

5. Dostupná z <http://gluonhq.com/open-source/scene-builder/>

a spôsob jeho zostavenia do výsledného archívu. Využiť je možné rôzne zásuvné moduly, tzv. *pluginy*, ktoré upravujú proces kompilácie a zabalenia tried. Väčšina externých knižníc použitých v projekte je uložená v centrálnom repozitári⁶ a do projektu sa pridajú iba ako položka v XML súbore. Ostatné knižnice je možné pridať lokálne. Pri kompilácii sa všetky závislosti stiahnu a zabalia do výsledného archívu.

Z ďalších knižníc je využívaná voľne dostupná knižnica **Google Guice 4.0**⁷, ktorá poskytuje podporu pre tzv. *vkładanie závislostí* – dependency injection pomocou anotácií. **Dependency injection** je v objektovo orientovanom programovaní technika uplatňujúca návrhový vzor **Inversion of Control** (IoC) [8], ktorý v princípe umožňuje prácu jedného programového komponentu so závislosťami, ktoré využíva, bez toho, aby na ne mal v dobe prekladu konkrétny odkaz. Pomocou IoC sa vložia konkrétne závislosti do komponentov až za behu programu. Technika sa realizuje vytváraním rozhraní pre jednotlivé komponenty a ich implementáciou, čo umožňuje lepšiu modularitu a rozširiteľnosť samotného programu.

Nové objekty, vylepšené kolekcie a pomocné triedy prináša voľne dostupná knižnica **Google Guava 18.0**⁸, ktorá je v programe taktiež použitá.

Na testovanie jednotlivých komponentov je použitá aplikačná nadstavba **JUnit Framework**⁹ vo verzii 4, ktorá umožňuje písanie jednotkových testov. Poskytuje anotácie `@Test` alebo `@Rule` a základné assert metódy.

Program bol vyvíjaný v komerčnom integrovanom vývojovom prostredí **IntelliJ IDEA**¹⁰, ktoré je pre univerzitné a študijné účely dostupný po registrácii na rok zadarmo.

Ako úložisko zdrojových kódov a zároveň verzovací nástroj bol použitý privátny repozitár na serveri **Bitbucket.org**¹¹, ktorý využíva voľne dostupný distribuovaný systém na správu revízií – **git**¹².

6. The Central Repository – dostupný na <http://central.sonatype.org/>

7. Domovská stránka projektu na <https://github.com/google/guice>

8. Domovská stránka projektu na <https://github.com/google/guava>

9. Domovská stránka projektu na <http://junit.org/>

10. Domovská stránka programu na <https://www.jetbrains.com/idea/>

11. Dostupný na adrese <https://bitbucket.org/>

12. Domovská stránka projektu na <https://git-scm.com/>

Na riadenie a rozvrhovanie vývoja bola použitá webová aplikácia určená na správu projektov a projektový manažment – **Trello**¹³. Aplikácia umožňuje vytváranie kariet s jednotlivými úlohami, ich prehľadné radenie do listov a tabúl, označovanie štítkami, plánovanie doby trvania a pod.

4.1 Dostupné knižnice na prácu s FITS

Na prácu s FITS súbormi a ich úpravu je dostupných vyše 30 knižníc do rozličných programovacích jazykov [9]. Do jazyka Java sú dostupné nasledovné 4 knižnice:

- **STIL – Starlink Tables Infrastructure Library**
Knižnica určená na spracovanie tabuľkových dát z rôznych súborových formátov. Okrem FITS tabuliek podporuje VOTable tabuľky, SQL tabuľky, CSV súbory a ďalšie. Knižnica je voľne dostupná pod licenciou LGPL¹⁴, je v aktívnom vývoji a jej zdrojové kódy sú prístupné v repozitári na serveri GitHub¹⁵. Keďže dokáže spracovávať len ASCII a binárne FITS tabuľky, je pre nástroj **nepoužiteľná**.
- **eap.fits**
Knižnica dokáže čítať a zapisovať FITS súbory, poradí si ako s obrazovými, tak aj tabuľkovými dátami a ich hlavičkami. Výhodu má v triede `RandomAccessFitsFile`, ktorá súbor nenačítava do pamäte, ale pristupuje k nemu pomocou ukazovateľov, takže si načíta len tú časť, ktorú aktuálne potrebuje. Knižnica obsahuje aj jednoduchú aplikáciu na prehliadanie a úpravu FITS súborov. Dokáže čítať súbory zašifrované pomocou PGP¹⁶, ak jej je poskytnutá príslušná rozširujúca knižnica. Zdrojové kódy má prístupné ako JAR archív na stránkach NASA¹⁷. Podľa dokumentácie je jej posledná verzia **1.3** z roku 2004, ďalej sa nevyvíja a preto je na použitie **nevhodná**.

13. Domovská stránka aplikácie na <https://trello.com/>

14. GNU Lesser General Public License – open-source licencia

15. <https://github.com/Starlink/starjava>

16. Pretty Good Privacy – spôsob šifrovania dát založený na štandarde OpenPGP

17. <http://swift.gsfc.nasa.gov/sdc/software/java/>

- **jfits**

Knižnica vznikla ako súčasť systému VLT Data Flow System¹⁸, ktorý sa používa na spracovanie veľkého množstva dát na sústave teleskopov v Chile organizácie *Európske južné observatórium*. Poskytuje základné operácie s hlavičkami FITS súborov (pridanie, odstránenie, iterácia a i.). Podľa oficiálnej stránky¹⁹ je knižnica v rannom štádiu vývoja, tzv. alpha verzii, a má niekoľko funkčných obmedzení. Zdrojové kódy sú voľne dostupné pod licenciou GNU GPL v2²⁰ ako archív z oficiálnej stránky. Aktuálne posledná verzia **0.94** je z mája 2008 a podľa všetkého sa ďalej nevyvíja ani neudržiava, preto takisto **nie je vhodná** na použitie.

- **nom.tam.fits** Knižnica bola vytvorená s účelom efektívneho čítania a zapisovania FITS súborov v jazyku Java. Podporuje všetky základné FITS formáty, ASCII aj binárne tabuľky a dokáže pracovať aj s komprimovanými súbormi. Je v aktívnom vývoji, v oficiálnom repozitári na serveri GitHub²¹ pravidelne pribúdajú nové aktualizácie zdrojového kódu, aktuálna verzia **1.13.1** je z 21. augusta 2015. Knižnica je voľne dostupná pod verejnou licenciou a uložená v centrálnom repozitári⁶, takže je ľahko použiteľná so systémom Apache Maven [7]. Oproti ostatným knižniciam má výhodu vo väčšej variabilite operácií s hlavičkovými záznamami a podporuje záznamy s dlhými textovými refazcami. Práve kvôli uvedeným dôvodom je táto knižnica **použitá** v oboch častiach programu.

18. Podrobnosti o systéme na <https://www.eso.org/projects/dfs/>

19. Dostupná na http://www.eso.org/~pgrosbol/fits_java/jfits.html

20. GNU General Public License version 2 – open-source licencia

21. <https://github.com/nom-tam-fits/nom-tam-fits>

5 Implementácia

Ako už bolo špecifikované v požiadavkách (kapitola 2), celý nástroj sa skladá z dvoch nezávislých častí – **výkonnej** a **grafickej**. Obe sú samostatne spustiteľné, výkonná časť z príkazového riadku, grafická časť ako desktopová aplikácia. Distribuované sú ako štandardný **JAR** archív. Môžu byť využité nezávisle, grafická časť ale volá tú výkonnú, takže k nej potrebuje mať špecifikovanú cestu.

Toto oddelenie oboch častí má niekoľko výhod. V prvom rade, výkonná časť môže byť spúšťaná samostatne v systémoch, kde nie je dostupné grafické prostredie. Za druhé, v prípade, že by obe časti boli implementované ako jedna, mohlo by sa pri výskyte chyby vo výkonnej časti stať celé grafické rozhranie aplikácie neresponzívne, teda mohlo by „zamrznúť“, čo nie je pre koncového užívateľa príjemné a želané chovanie. Takto môže mať aplikácia lepšiu kontrolu nad výkonnou časťou. V neposlednom rade je oddelená grafická časť viac prenositeľná, čo znamená, že je možné zameniť výkonnú časť za inú s minimálnymi úpravami v grafickej aplikácii.

Nasledujúce podkapitoly sa venujú podrobnému popisu oboch častí a ich implementačným detailom.

5.1 Finálny návrh hlavičkových operácií

Pred návrhom architektúry bolo potrebné stanoviť, ktoré hlavičkové operácie bude program finálne poskytovať. V kombinácii s požiadavkami zo zadania a operáciami z fotometrického archívu¹ bolo určených týchto 8 operácií + 2 dodatočne po rozšírení požiadaviek:

- **pridať nový záznam na koniec hlavičky**
- **pridať nový záznam na konkrétny index v hlavičke**
- **odstrániť záznam z hlavičky podľa kľúčového slova**

1. Fotometrický archív[10] bol vytvorený ako súčasť diplomovej práce v spolupráci s Ústavom teoretickej fyziky a astrofyziky PrF MU a jeho časťou bola normalizácia FITS hlavičiek pred tým, než sa uložia do samotného archívu. Keďže sa používa astronómami, uvedené operácie by pre nich mohli byť vhodné aj v tomto programe.

- odstrániť záznam z indexu v hlavičke
- zmeniť kľúčové slovo hlavičkového záznamu
- zmeniť hodnotu hlavičkového záznamu
- zrefaziť viacero hlavičkových záznamov
- posunúť čas časového záznamu hlavičky
- vypočítať Juliánsky dátum
- vypočítať Heliocentrický Juliánsky dátum

5.2 Výkonná časť

Programovací jazyk Java člení jednotlivé triedy do väčších ucelených celkov – **balíkov**.

Výkonná časť je uložená v základnom balíku `cz.muni.fi.fits`. V ňom sa nachádzajú menšie balíky, ktoré združujú komponenty s rovnakým funkčným zameraním.

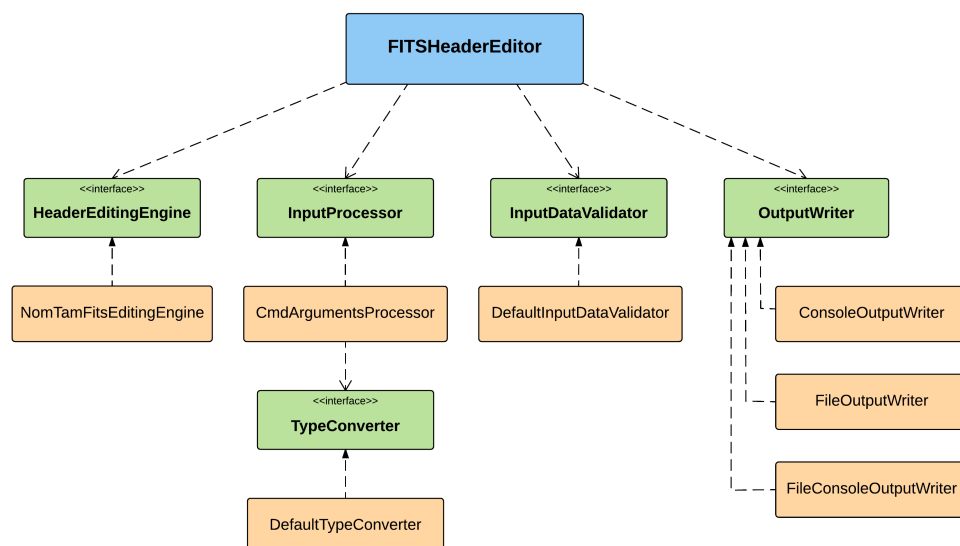
Jednotlivé komponenty sú písané formou rozhraní a ich implementácií. Rozhrania v objektovom programovaní definujú predpísaný kontrakt, ktorý budú všetky implementujúce triedy poskytovať. To znamená, že každá taká trieda musí implementovať všetky deklarované metódy. Komponent potom bude pracovať len s metódami rozhrania a nebude závislý na konkrétnej implementácii, čo umožňuje modularitu programu a zameniteľnosť jednotlivých častí za iné, podľa potreby.

Architektúra programu je rozdelená na jednu hlavnú časť a štyri menšie, zato funkčne bohatšie, celky.

Základnou časťou je trieda ***FITSHeaderEditor***. Obsahuje verejný konštruktor prijímajúci všetky potrebné závislosti. Sú to štyri základné rozhrania definujúce vyššie spomenuté celky: *InputProcessor*, *InputDataValidator*, *HeaderEditingEngine* a *OutputWriter*. Túto situáciu zachytáva diagram tried na obrázku 5.1.

Okrem neho trieda obsahuje verejnú metódu `start()`. V nej prebieha najvyššia abstrakcia úpravy hlavičiek. Metóda používa prijaté závislosti a pomocou nich spracúva vstupné argumenty, overuje ich

správnosť a aplikuje ich na hlavičky FITS súborov, pričom na výstup zaznamenáva správy z týchto operácií.



Obr. 5.1: Diagram hlavných tried výkonnej časti

5.2.1 Spracovanie vstupov

Pred samotným popisom spracovania je potrebné určiť spôsob, akým sa budú vstupné dáta programu predávať, a ich formát.

Program bude vstupné dáta prijímať z **príkazového riadku**. Keďže má upravovať hlavičky viacerých súborov naraz, nebolo by užívateľsky príjemné zadávať jednotlivé cesty k FITS súborom oddelene. Navyše, operačné systémy majú limitovanú dĺžku argumentov použitých v príkazovom riadku. V systémoch MS Windows je to 8192 bajtov [11], v Linuxových systémoch je maximálna dĺžka premenlivá (je možné ju zistiť pomocou príkazu `xargs --show-limits`), väčšinou je limit okolo 2 MB. Kvôli uvedeným dôvodom je potrebné cesty k FITS súborom na úpravu vložiť do jedného súboru (tzv. **vstupného súboru**), ktorého cesta sa predá ako argument na vstupe.

Popis vstupného súboru Vstupný súbor obsahuje buď priame cesty k FITS súborom, ktorých hlavičky majú byť upravené, alebo cestu

k priečinku so súbormi, alebo oboje naraz. Každá cesta musí byť zapísaná na samostatnom riadku, oddelená znakmi nového riadku (ASCII znaky 10 a 13 decimálne). Jednotlivé cesty môžu byť zakomentované znakom # (ASCII 35 decimálne), v tom prípade budú preskočené.

Ukážka vstupného súboru:

```
C:\Dokumenty\FITS súbory\FITS_001.fits
C:\Dokumenty\FITS súbory\FITS_002.fits
# C:\Dokumenty\FITS súbory\FITS_003.fits
D:\Projekty\Obrázky\FITS
```

Operácie majú pri predávaní argumentov jednotný formát:

OPERÁCIA **▯ prepínače** **▯ vstupný súbor** **▯ argumenty operácie**

Podrobnejší popis argumentov pre jednotlivé operácie sa nachádza v prílohe 1.

Prijímaním a spracovaním vstupov do programu sa zoberá rozhranie *InputProcessor*. Je navrhnuté tak, aby do programu prijalo vstupy s údajmi o operácii a súboroch (z príkazového riadku, zo súboru, zo vzdialeného servera atď.) a poskytlo ich v spracovanej forme pomocou metódy *getProcessedInput()*. Rozhranie implementuje trieda ***CmdArgumentsProcessor*** a spracúva vstupné argumenty z príkazového riadku. Kvôli lepšej štruktúre v hlavnej metóde využíva pomocnú triedu ***CmdArgumentsProcessorHelper***. Tá obsahuje 10 metód na spracovanie vstupných dát podľa typu operácie a jednu metódu, ktorá načítava FITS súbory zo vstupného súboru.

Keďže všetky argumenty sú načítané ako textové reťazce, je potrebné ich konvertovať do odpovedajúcich typov. Na to slúži komponent ***TypeConverter*** (rozhranie), ktorý trieda *CmdArgumentsProcessor* dostane v konštruktore. Ten obsahuje metódy na analýzu a konverziu reťazca do niekoľkých základných typov: *Integer*, *Long*, *Double*, *Boolean*, *BigInteger*, *BigDecimal* a *LocalDateTime*. Rozhranie implementuje trieda ***DefaultTypeConverter***.

V metóde *getProcessedInput()* sa spracované a pretypované argumenty uložia do objektu triedy ***InputData***. Je to abstraktná trieda obsahujúca atribút enumeračného typu ***OperationType***, ktorý ur-

čuje operáciu, akú objekt reprezentuje. Každá hlavičková operácia má samostatnú triedu, ktorá ju priamo alebo nepriamo rozširuje a obsahuje ďalšie atribúty špecifické pre danú operáciu. Pri operáciách, kde sú využívané prepínače, triedy rozširujú abstraktnú triedu *SwitchInputData* (a tá priamo rozširuje triedu *InputData*).

Z metódy prijatý objekt sa vo *FITSHederEditor*-e pošle do ďalšej časti, kde sa overí správnosť vstupných dát.

5.2.2 Validácia vstupných dát

Slúži na overovanie korektnosti vstupných dát – či obsahujú všetky potrebné údaje, či sa v nich nevyskytujú nepovolené znaky, či spĺňajú maximálnu dĺžku a pod.

Hlavným komponentom je rozhranie *InputDataValidator*, ktoré obsahuje metódu *validate()* bez návratovej hodnoty a preťaženie v 10-tich verziách podľa daného typu operácie. Na vstupe prijíma objekt konkrétneho typu triedy *InputData* a ak sú všetky jeho dáta v správnom formáte, funkcia prebehne bez vyhodenia výnimky. Ak je však niektorý údaj nesprávny, funkcia vyhodí výnimku *ValidationException* s popisom konkrétneho problému. To znamená, že vstupné dáta nie sú v správnom formáte a je potrebné ich zadať znova.

Rozhranie implementuje trieda *DefaultInputDataValidator*, ktorej inštanciu dostane *FITSHederEditor* v konštruktoze.

V metódach sa kontrolujú nasledovné veci:

- sú zadané všetky povinné údaje k danej operácii
- vo vstupnom súbore sa nachádza aspoň jedna cesta k súboru alebo priečinku
- správna dĺžka kľúčového slova, hodnoty, komentára alebo hodnoty a komentára dokopy
- sú použité iba štandardu vyhovujúce ASCII znaky
- čísla, dátumy a časy sú zadané v správnom formáte

Trieda využíva okrem iného aj regulárne výrazy, napríklad na overovanie korektnosti kľúčového slova.

Ak je validácia úspešná, vstupné dáta sa vo *FITSHederEditor*-e predajú do ďalšej časti, kde už prebieha samotná úprava hlavičiek.

5.2.3 Úprava hlavičiek zadaných súborov

Najdôležitejšia časť, kde sa priamo pristupuje k FITS súborom a upravuje sa ich hlavička. Definuje ju rozhranie *HeaderEditingEngine*, ktoré implementuje trieda *NomTamFitsEditingEngine* s názvom podľa použitej knižnice **nom.tam.fits** (4.1).

V tejto časti je možné vidieť výhodu použitia rozhraní. Konkrétne rozhranie je totiž možné implementovať s použitím iných FITS knižníc (v budúcnosti výkonnejších a pod.) a zamieňať ich bez problémov a zásahov v hlavnej triede *FITSHeaderEditor*.

Rozhranie predpisuje 10 verejných metód podľa jednotlivých operácií, ktoré na vstupe prijímajú odpovedajúce dáta. Každá metóda vykonáva danú operáciu nad jedným FITS súborom. Jej postup je viac-menej nasledovný:

1. Otvorí a načíta FITS súbor do pamäte a prečíta hlavičku prvej HDU jednotky.
2. Skontroluje, či je možné vstupné dáta aplikovať do konkrétnej hlavičky.
3. Vykoná úpravu hlavičky podľa zadaných dát.
4. Uloží zmeny späť do súboru a zavrie ho.
5. Vráti výsledok operácie ako objekt triedy *Result*.

Metódy majú návratový typ *Result* obsahujúci atribút *success* typu *boolean*, ktorý indikuje či sa operácia skončila úspešne alebo nie. V druhom atribúte *message* sa potom nachádzajú podrobnosti k výsledku operácie.

Keďže FITS hlavička obsahuje niekoľko povinných záznamov, v každej metóde sa kontroluje, či do nich operácia nezasahuje a nenaťahuje ich formát – ak by sa takéto záznamy nevhodne upravili, FITS súbor by sa mohol stať nečitateľným, preto je akákoľvek ich úprava zakázaná. Slúži na to statická trieda *MandatoryFITSKeywords* s metódou *matchesMandatoryKeyword()*, ktorá porovnáva poskytnuté kľúčové slovo s tými povinnými a vráti *true*, ak kľúčové slovo medzi ne patrí, *false* v opačnom prípade.

Niektoré metódy obsahujú vstupný parameter všeobecného typu *Object*. Je to kvôli tomu, že v ňom môžu prijímať dáta rôzneho typu:

- metóda `addNewRecord(..., Object value, ...)`
Hodnota novo-pridávaného záznamu musí mať jeden z nasledovných typov: *Integer*, *Long*, *Double*, *Boolean*, *String*, *BigInteger* alebo *BigDecimal*. V prípade iných typov metóda skončí neúspešne.
- metóda `addNewRecordToIndex(..., Object value, ...)`
Rovnaký prípad ako vyššie uvedená metóda `addNewRecord()`.
- metóda `changeValueOfRecord(..., Object newValue, ...)`
Opäť rovnaký prípad, nová hodnota existujúceho záznamu musí mať jeden z vyššie uvedených typov.
- metóda `computeJulianDate(Object datetime, Object exposure, ...)`
Pri výpočte Juliánskeho dátumu je možné hodnotu dátumu a času pozorovania získať dvojako:
 - zadaním kľúčového slova existujúceho záznamu s časovou hodnotu – typ *String*,
 - priamym zadaním dátumu a času v prípade, že hlavička podobný záznam neobsahuje – typ *LocalDateTime*.

Podobne aj pri parametri expozičného času. Ten je možné zadať buď ako kľ. slovo existujúceho záznamu (typ *String*), alebo priamo ako hodnotu v sekundách vo forme desatinného čísla (typ *Double*). Pri iných typoch metóda skončí neúspešne.
- metóda `computeHeliocentricJulianDate(Object datetime, Object exposure, Object rightAscension, Object declination, ...)`
Pri parametroch `datetime` a `exposure` platí to isté, čo pri výpočte Juliánskeho dátumu.
Parameter `rightAscension`, teda **rektascenzia**², je možné zadať tromi spôsobmi:

2. súradnica udávajúca **uhol**, ktorý zviera rovina deklinačnej kružnice hviezd s rovinou deklinačnej kružnice prechádzajúcej jarným bodom, viac na https://en.wikipedia.org/wiki/Right_ascension

- zadáním kl. slova záznamu, ktorý sa už nachádza v hlavíčke (typ *String*)
- zadáním desatinného čísla, ktoré vyjadruje hodnotu (typ *Double* alebo *BigDecimal*)
- zadáním hodnoty v plnom formáte – v hodinách, minútach a sekundách (typ *TimeObject* – trieda vytvorená na časové hodnoty)

Parameter *declination*, teda **deklinácia**³, je takisto možné zadať trojako:

- zadáním kl. slova záznamu, ktorý sa už nachádza v hlavíčke (typ *String*)
- zadáním hodnoty vo forme desatinného čísla (typ *Double* alebo *BigDecimal*)
- zadáním hodnoty v plnom formáte – stupňoch, minútach a sekundách (typ *DegreesObject* – trieda vytvorená na stupňové hodnoty)

Ak je do metódy predaný iný typ hodnoty, metóda skončí neúspechom.

Okrem hlavných metód obsahuje trieda *NomTamFitsEditingEngine* niekoľko privátnych metód, ktoré obsahujú viackrát využívané časti kódu – predchádza sa tak jeho zbytočnej duplikácii.

Na výpočet Juliánskeho dátumu a Heliocentrického Juliánskeho dátumu slúžia dve triedy:

- ***JulianDate***

Obsahuje statické metódy na výpočet Juliánskeho dátumu. Výstupom je **desatinné číslo**. Čas a dátum prijímajú buď ako objekt typu *LocalDateTime*, alebo jednotlivo po časových jednotkách – rok, mesiac, deň, hodiny, minúty, sekundy a nanosekundy.

3. súradnica udávajúca **uhlovú vzdialenosť** hviezdy od svetového rovníka meraná pozdĺž deklináčnej kružnice hviezdy, viac na <https://en.wikipedia.org/wiki/Declination>

- ***HeliocentricJulianDate***

Obsahuje statické metódy na výpočet Heliocentrického Juliánskeho dátumu. Výstupom je taktiež desatinné číslo. Na jeho výpočet je potrebné dodať:

- **Juliánsky dátum** – buď ako **desatinné číslo** (typ *Double*), alebo ako dátum a čas (typ *LocalDateTime*), z ktorého sa dopočíta Juliánsky dátum,
- **rektascenziu** – buď ako hodnotu vo forme desatinného čísla (typ *Double*), alebo ako objekt typu *TimeObject* s jednotlivými zložkami hodnoty
- **deklináciu** – buď ako hodnotu vo forme desatinného čísla (typ *Double*), alebo ako objekt typu *DegreesObject* s jednotlivými zložkami hodnoty

Algoritmus výpočtu oboch hodnôt bol použitý zo špecializovanej stránky univerzity **Stephen F. Austin State University** [12]

5.2.4 Spracovanie výstupu z operácií

Po uskutočnení operácie sa výsledok pošle do poslednej časti, ktorá sa stará o spracovanie a výpis všetkých správ, ktoré program produkuje. Túto časť tvorí rozhranie ***OutputWriter***. Predpisuje niekoľko metód na spracovanie výstupu, podľa typu výsledku. V základe obsahuje tri druhy metód, ktoré sú ďalej preťažené pre rôzne typy vstupných parametrov:

- **`writeInfo()`**
Vypisuje **informačné správy**, napr. po úspešnom vykonaní operácie.
- **`writeError()`**
Vypisuje **chybové správy**, napr. ak operácia nebola úspešná.
- **`writeException()`**
Vypisuje **správy z výnimiek**, napr. chyby pri validácii vstupov a pod.

Toto rozhranie implementujú až 3 triedy, podľa toho, kde vypisujú výstupné správy:

- ***ConsoleOutputWriter***
Správy sú vypisované priamo do konzoly, z ktorej je program spúšťaný.
- ***FileOutputWriter***
Správy sú vypisované do súboru, ktorý objekt dostane v konštruktoze.
- ***FileConsoleOutputWriter***
Správy sú vypisované aj do konzoly, aj do súboru špecifikovaného v konštruktoze.

Rozhranie môže byť v budúcnosti implementované napríklad triedou, ktorá bude výpisy ukladať na vzdialený server a pod.

Trieda *FITSHederEditor* môže počas behu používať len jednu z týchto tried. Komponenty je možné medzi behmi programu jednoducho vymieňať, bez re-kompilácie celého programu. Slúži na to konfiguračný súbor *fits.properties*, ktorý sa načítava počas spúšťania programu. Je **povinný** a musí byť vždy dostupný v rovnakom priečinku ako spúšťaný program. Pomocou dvojíc kľúč–hodnota sa v ňom dá nastaviť typ výstupu nasledovne:

- výpis **do konzoly** – súbor musí obsahovať toto nastavenie:

```
output.writer = console
```

- výpis **do súboru** – súbor musí obsahovať aj cestu k výstupnému súboru:

```
output.writer = file  
output.file = output.txt
```

- výpis **do konzoly a súboru** zároveň – súbor musí obsahovať aj cestu k výstupnému súboru:

```
output.writer = file, console  
output.file = output.txt
```

Prípadné komentáre je možné písať na samostatný riadok, ktorý začína znakom #.

Celý proces od zadania vstupných argumentov, cez validáciu a úpravu, až po výpis na výstup, je zachytený v sekvenčnom diagrame 5.2, ktorý proces znázorňuje prehľadne graficky.

5.2.5 Ostatné časti

Ako už bolo spomenuté v kapitole 4 o použitých technológiách, výkonná časť využíva princíp **Inversion of Control**, konkrétne techniku *vkładania závislostí* (**dependency injection**) pomocou externej knižnice **Google Guice**.

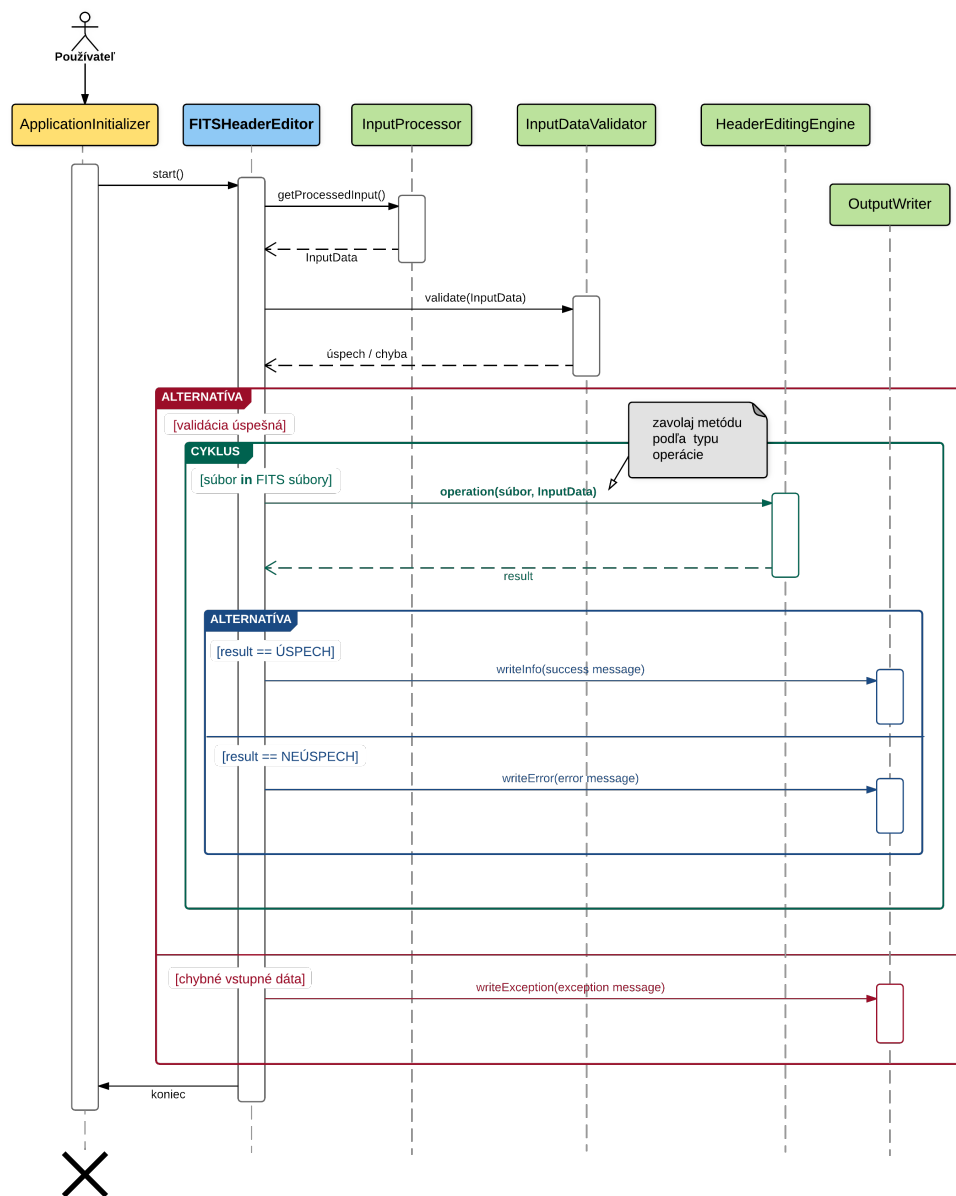
Základným krokom je rozšírenie triedy *AbstractModule* z balíka `com.google.inject` a implementovanie metódy `configure()`, v ktorej sa nakonfigurujú potrebné závislosti. V nej sa k rozhraniám naviažu konkrétne triedy (ktoré implementujú dané rozhranie), takže keď knižnica „uvidí“, že komponent využíva dané rozhranie, vloží do neho inštanciu tejto triedy. Náhľad kódu je z triedy *AppInjector*, ktorá rozširuje *AbstractModule*:

```
bind(HeaderEditingEngine.class)
    .to(NomTamFitsEditingEngine.class);
bind(InputDataValidator.class)
    .to(DefaultInputDataValidator.class);
```

O samotné vytvorenie objektov sa už postará knižnica. Tá podľa konfigurácie hľadá všetky triedy označené anotáciou `@Singleton` (napríklad trieda *NomTamFitsEditingEngine*) a vloží ich inštancie do tried, ktoré majú konštruktory alebo metódy označené anotáciou `@Inject` (v tomto prípade hlavná trieda *FITSHheaderEditor*). Náhľad kódu je z triedy *ApplicationInitializer*:

```
// inject all necessary dependencies
Injector injector =
    Guice.createInjector(new AppInjector(...));

// get instance of executive class
FITSHheaderEditor editor =
    injector.getInstance(FITSHheaderEditor.class);
```



Obr. 5.2: Sekvenčný diagram práce výkonnej časti

Knižnica vytvorí objekt triedy *FITSHeaderEditor* so všetkými potrebnými závislosťami, takže na ňom stačí zavolať metódu `start()`:

```
// start FITS header editing operation
editor.start();
```

V triede *ApplicationInitializer* prebieha aj načítanie konfiguračného súboru *fits.properties* do objektu typu *Configuration*. Ten je predaný do triedy *AppInjector*, kde sa podľa neho určí inštancia výstupnej triedy.

5.2.6 Spustenie programu

V neposlednom rade je potrebné spomenúť spôsob, akým sa výkonná časť spúšťa z príkazového riadka.

V systéme je potrebné mať nainštalované behové prostredie jazyka Java – **Java Runtime Environment** vo verzii 8 a vyššej⁴.

Spustenie prebieha zadáním nasledujúcich príkazov do príkazového riadka spusteného z priečinka s programom:

```
java -jar cesta k JAR súboru OPERÁCIA parametre
```

5.3 Grafická časť

Grafická časť nástroja sa nachádza v balíku *cz.muni.fi.fits.gui* a jednotlivé komponenty má ďalej rozdelené do menších balíkov.

Napísaná je v jazyku Java s využitím zabudovanej grafickej knižnice **JavaFX 8**. Je spustiteľná ako samostatná aplikácia, na úpravu hlavičiek ale potrebuje výkonnú časť.

Všetky komponenty v jazyku JavaFX sa skladajú z dvoch základných častí:

- **FXML súbor**

Je to v princípe XML súbor s vlastnými značkami, ktorý definuje samotný vzhľad aplikácie – rozloženie ovládacích prvkov, veľkosť, farbu a i. Každá grafická súčasť je definovaná svojím FXML súborom.

4. dostupné z <http://java.com/en/download/>

- **riadiaca trieda** (controller)

Obsahuje hlavnú logiku komponentu, reaguje na udalosti (tzv. *events*) od užívateľa, prípadne iných komponentov, stará sa o zobrazovanie a správu dát na grafickej strane. Je to klasická Java trieda, ktorá je ale prepojená s FXML súborom pomocou špeciálneho XML atribútu. Nemusí byť vždy prítomná, najmä ak grafická časť slúži len na zobrazovanie statických dát bez ďalších interakcií.

Grafické rozhranie je možné navrhovať „naslepo“, teda písať jednotlivé časti a komponenty priamo do FXML súboru, alebo jednoduchšie s využitím grafického prostredia (tzv. *designer*), v ktorom sa dajú komponenty zostaviť vizuálne pomocou myši. FXML súbor sa potom vygeneruje automaticky. Pri JavaFX sa používa nástroj **SceneBuilder** a bol využitý aj pri návrhu rozhrania vyvíjaného nástroja.

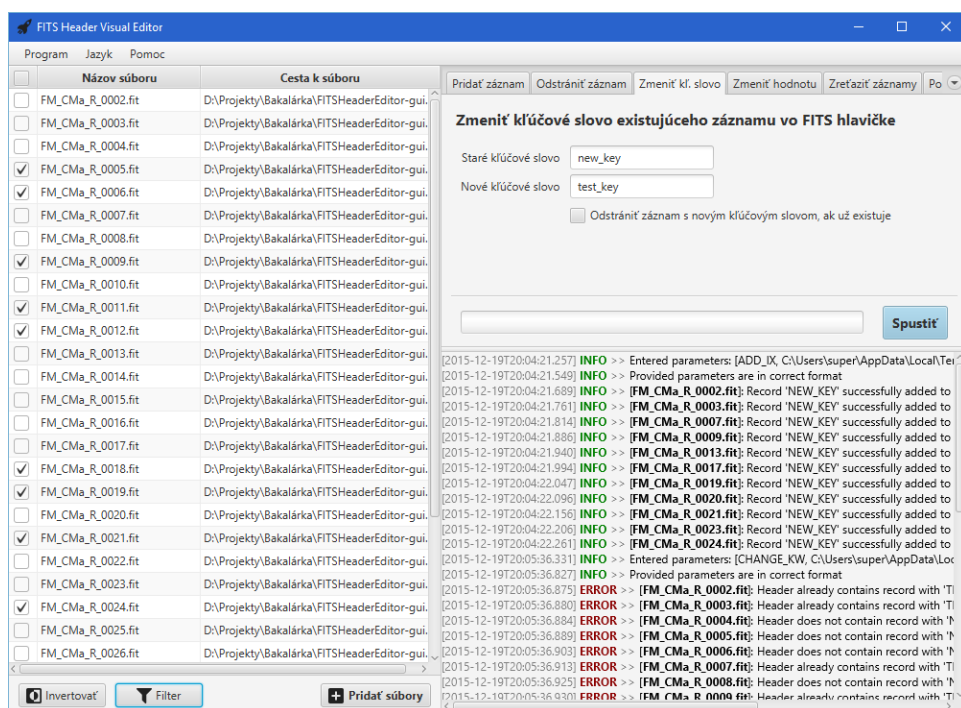
Aplikácia obsahuje jediné hlavné okno (na obrázku 5.3), ktoré je logicky rozdelené na 3 časti:

- **zoznam FITS súborov** – tvorí ľavú časť okna aplikácie a slúži na správu súborov, na ktorých bude vykonaná operácia
- **panely s operáciami** – tvoria hornú pravú časť aplikácie a obsahujú polia na zadanie parametrov špecifických pre danú operáciu
- **textový výstup** – spodná pravá časť aplikácie slúžiaca na výpis textových správ z výkonného programu

Po spustení vybranej operácie sa zadané parametre z panelu spolu s vybranými súbormi zo zoznamu spracujú do požadovaného formátu a odošlú ako parametre na vstup do výkonného programu. Výstup z neho sa potom zobrazuje v časti textového výstupu.

Okrem hlavného okna sú v aplikácii menšie **dialógové okná** (filter súborov, nastavenia programu a i.) a **okná s upozorneniami** (chyba pri zadávaní parametrov, úspešne dokončená operácia a i.).

Centrálnym komponentom grafického rozhrania programu je trieda **MainApp**. V nej sa načítavajú všetky ostatné komponenty a súčasti, ako sú FXML súbory či riadiace triedy, slúži tiež ako komunikačné spojenie medzi rôznymi časťami aplikácie. Rozširuje triedu *Application* z balíka `javafx.application`, ktorá je základným bodom



Obr. 5.3: Hlavné okno aplikácie

pri tvorbe grafických aplikácií v JavaFX a predstavuje jadro celej aplikácie.

V základnom balíku JavaFX sa tiež nachádza rozhranie *Initializable* s metódou `initialize(URL location, ResourceBundle resources)`. Používa sa v prípade, keď je aplikácia lokalizovaná do viacerých jazykov. Riadiaca trieda, ktorá implementuje toto rozhranie, potom cez parameter `resources` uvedenej metódy dostane pri inicializácii objekt s reťazcami použitých lokalizácií a tie sa automaticky podľa kľúčov priradia k jednotlivým komponentom. Objekt zároveň môže slúžiť aj na manuálne načítavanie reťazcov neskôr po inicializácii.

Keďže je aplikácia lokalizovaná do troch jazykov, všetky riadiace triedy implementujú toto rozhranie. Nie však priamo, ale cez pomocnú abstraktnú triedu *Controller*, ktorá združuje ich spoločnú logiku a komponenty. Tá si v inicializačnej metóde uloží objekt s reťaz-

cami do lokálnej premennej na neskoršie použitie. Obsahuje aj odkaz na centrálnu triedu *MainApp*, ktorú niektoré riadiace triedy využívajú.

5.3.1 Zoznam FITS súborov

Ľavá časť aplikácie slúži na správu načítaných FITS súborov a výber tých, ktorým bude upravovaná hlavička. Tvorí ju tabuľka s popisom súborov (JavaFX element *TableView*) a ovládacie tlačidlá naspodku.

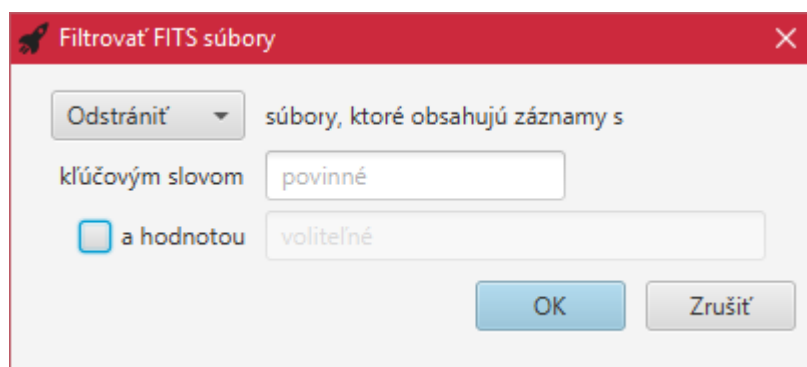
Na pridanie FITS súborov do tabuľky slúži tlačidlo **Pridať súbory**. Po kliknutí sa otvorí štandardný dialóg na načítanie súborov (JavaFX trieda *FileChooser*). Grafická stránka je v každom operačnom systéme odlišná, funkčne sú však dialógy rovnaké. Umožňuje vyhľadávať súbory podľa názvu (s použitím znakov **?*) a vyberať viacero súborov naraz. Obsahuje filter na súborové prípony *.fit*, *.fits* a *.fts*, takže načíta len validné FITS súbory. Po potvrdení výberu sú súbory uložené do objektu triedy *FileItem* a vložené do špeciálnej kolekcie *ObservableList*, ktorá dokáže automaticky synchronizovať položky s grafickou časťou. To znamená, že ak je do kolekcie pridaný nový objekt so súborom, automaticky sa zobrazí v tabuľke ako nový riadok, bez manuálneho volania metód na obnovenie náhľadu.

Trieda *FileItem* predstavuje jeden riadok tabuľky. Okrem odkazu na načítaný FITS súbor obsahuje aj ďalšie premenné (názov, cesta k súboru a i.), ktoré sú priamo naviazané na stĺpce v tabuľke, takže po pridaní nového objektu do kolekcie súborov sa ich hodnoty automaticky priradia do odpovedajúcich stĺpcov.

Začiarkavacie políčka v prvom stĺpci slúžia na označenie súborov, na ktorých bude vykonaná úprava hlavičiek. Umožňujú druhotný výber súborov z tabuľky. Zadaná operácia sa vždy vykoná iba na súboroch, ktoré sú začiarované. Tlačidlo **Invertovať** otočí začiarovanie na každom súbore na opačnú hodnotu, takže začiarované súbory budú nezačiarknuté a naopak.

Kliknutím pravého tlačidla myši na súbory v tabuľke sa zobrazí **kontextová ponuka**, ktorá umožní odstrániť vybraný súbor alebo súbory z tabuľky alebo zmeniť jeho/ich začiarovanie.

Súbory v tabuľke je možné aj filtrovať podľa hlavičkových záznamov, ktoré obsahujú. Po kliknutí na tlačidlo **Filter** sa zobrazí nasledovný dialóg:



Obr. 5.4: Dialóg na nastavenie filtra FITS súborov

Filtrovať je možné buď podľa zadaného kľúčového slova, alebo podľa kombinácie kľúčového slova a hodnoty, ktorú obsahuje, pričom stačí, aby hodnota obsahovala iba zadaný text ako podreťazec. Vyhovujúce súbory sa v tabuľke buď ponechajú, alebo odstránia, podľa vybranej možnosti vo výberovom poli.

Samotný proces filtrácie prebieha ako úloha vo vedľajšom vlákne procesora, takže grafické rozhranie zostane aj naďalej responzívne. Úlohu predstavuje trieda *FilteringTask* (potomok zabudovanej triedy *Task* z balíka `javafx.concurrent`) a na čítanie záznamov v hlavičke FITS súborov sa používa podobne ako vo výkonnom programe knižnica **nom.tam.fits** (4.1).

5.3.2 Panely s operáciami

Pravú hornú časť tvoria panely s jednotlivými hlavičkovými operáciami. Každý panel obsahuje ovládacie prvky a polia špecifické pre danú operáciu.

Základnou riadiacou triedou je *OperationTabsViewController* s príslušným FXML súborom, ktorý definuje tlačidlo `Spustiť` a pás s postupom operácie. Tieto prvky sú spoločné pre všetky panely a sú spojené s procesom operácie. Trieda obsahuje aj kolekciu všetkých panelov (objektov riadiacich tried).

Každý panel tvorí samostatný FXML súbor a riadiaca trieda implementujúca spoločné rozhranie *TabController*. To definuje metódu

`getInputData()`, ktorá je volaná z vyššie uvedenej triedy a po spustení operácie serializuje užívateľom zadané dáta do jednotného objektu *InputData* (podobná štruktúra ako pri výkonnej časti programu).

Polia na zadanie vstupných dát operácie na paneloch sú kontrolované v reálnom čase pomocou regulárnych výrazov, to znamená, že užívateľovi nedovolí zadať neplatný znak, záleží od typu poľa. Napríklad pole pre kľúčové slovo dovoľuje zadať iba povolené ASCII znaky s maximálnou dĺžkou 8 znakov, podobne pre ostatné polia.

Pri operáciach, kde sa pracuje s hodnotou hlavičkového záznamu, sa rozlišujú tieto rôzne typy poľa:

- pole pre **textový reťazec** – len povolené ASCII znaky a číslice
- pole pre **číslo** – len číslice a desatinná bodka
- pole pre **logickú hodnotu** – začiarkavacie políčko
- pole pre **čas** - len číslice a znaky dvojbodka a bodka, korektný formát je HH:mm:ss.SSS
- pole pre **dátum** – výber myšou pomocou špeciálneho JavaFX komponentu *DatePicker*
- pole pre **čas spolu s dátumom** – kombinácia dvoch vyššie uvedených polí pre čas a dátum
- pole pre **konštantu** – pri operácii reťazenia záznamov, povolené sú opäť len validné ASCII znaky a číslice
- pole pre **plnú hodnotu** – platí len pri *deklinácii* a *rektascenzii* v operácii s Heliocentrickým Juliánskym dátumom, povolené sú len celé čísla z obmedzeným rozsahom
- pole pre **desatinné číslo** – platí opäť len pri *deklinácii* a *rektascenzii*, povolené sú len desatinné čísla z obmedzeným rozsahom

Podľa vybraného typu sa potom zadané dáta presne naformátujú na vstup výkonného programu.

Začiarkavacie políčka v paneloch predstavujú prepínače upravujúce chovanie danej operácie, presne ako prepínače vo výkonnej časti programu.

Po stlačení tlačidla `Spustiť` sa začnú spracovávať zadané dáta a spustí sa samotná operácia úpravy hlavičiek v nasledujúcich krokoch:

- **Overenie zadaných hodnôt**

Najskôr sa overí, či sú vyplnené všetky povinné polia pre danú operáciu a či sú zadané v správnom formáte – na to slúži vnútorná trieda *Validator*, ktorá sa nachádza v každej riadiacej triede a obsahuje statické metódy na overenie vstupov.

V prípade, že je hodnota v nesprávnom formáte, zobrazí sa dialóg s upozornením a operácia sa nevykoná.

- **Serializácia do objektu *InputData***

Po overení zadaných hodnôt sa tieto hodnoty zostavia do objektu triedy *InputData* a objekt sa predá do základnej riadiacej triedy *OperationTabViewController*.

- **Pridanie FITS súborov na úpravu**

Zo zoznamu FITS súborov sa vyberú označené položky a pridajú sa do kolekcie v objekte *InputData*. Ak nie sú vyznačené žiadne súbory, zobrazí sa upozornenie a spracovanie ďalej nebude pokračovať.

- **Zostavenie vstupných parametrov**

Z objektu *InputData* sa zostavia vstupné parametre do výkonnej časti podľa danej operácie a vytvorí sa dočasný súbor⁵ s cestami k FITS súborom.

- **Spustenie vykonávania hlavičkovej operácie**

Vytvorí sa špecializovaná úloha – objekt triedy *EditingTask*, do ktorej sa predajú vygenerované vstupné argumenty a ktorá zavolá výkonný program. Volanie externého programu sa v jazyku Java robí pomocou tried *Process* a *ProcessBuilder* nasledovným spôsobom:

5. umiestnenie dočasného súboru sa líši podľa operačného systému

```
// set up the process
Process process = new
    ProcessBuilder(arguments)
        .directory(workingDirectory.toFile())
        .redirectErrorStream(true)
        .start();

// do other tasks

// wait for process to finish
process.waitFor();
```

Úloha sa potom spustí vo vedľajšom vlákne na pozadí, aby neza-
blokovala celú aplikáciu. Počas vykonávania úlohy sa obnovuje
aktuálny postup operácie a graficky sa zobrazuje v hlavnom
okne v páse s postupom (tzv. *progress bar*).

Ak je operácia úspešne dokončená, zobrazí sa dialóg s ozná-
mením o úspešnom vykonaní, v opačnom prípade sa zobrazí
dialóg s popisom chyby.

Dočasný súbor s cestami k FITS súborom sa po ukončení ope-
rácie automaticky zmaže.

Štandardný a chybový výstup z výkonného programu sú pre-
smerované do dátového prúdu (objektu *InputStream*) a z neho
čítané a odosielané do časti s textovým výstupom.

5.3.3 Textový výstup

V tejto časti sa naformátuje a zobrazí textový výstup prijatý z výkon-
ného programu. Rovnako ako v ňom, aj tu sa rozlišujú tri druhy správ:

- **informačné správy** – zobrazia sa s označením INFO v zelenej farbe, nasledované samotnou správou
- **chybové správy** – označené slovom ERROR v červenej farbe
- **chybové správy s výnimkou** – označené tmavočerveným slo-
vom EXCEPTION s názvom výnimky a jej popisom

Hlavnou triedou je riadiaca trieda *OutputViewController*, v ktorej sa nachádzajú metódy na spracovanie udalostí so správami z výkonného programu.

Grafickú stránku v FXML súbore tvorí v podstate len jediný prvok – JavaFX komponent *TextFlow*, ktorý je najvhodnejší na zobrazovanie štruktúrovaného textu. S menšími úpravami dokáže automaticky zobrazovať najnovšiu správu naspodku podobným štýlom, aký je známy z príkazového riadku.

5.3.4 Lokalizácia

Grafická aplikácia je lokalizovaná do 3 jazykov: **slovenčiny, češtiny a angličtiny**.

Každý jazyk má vlastný *.properties* súbor z dvojicami **klúč – hodnota**, kde klúč je všeobecný identifikátor reťazca naprieč viacerými súbormi a hodnota obsahuje reťazec v danom jazyku.

Na lokalizáciu je určená trieda *ResourceBundle*, ktorá tieto súbory načítava a poskytuje preložené reťazce jednotlivým komponentom. Ako už bolo spomenuté na začiatku popisu grafickej časti, objekt tejto triedy je vložený do riadiacich tried pomocou metódy v rozhraní *Initializable*.

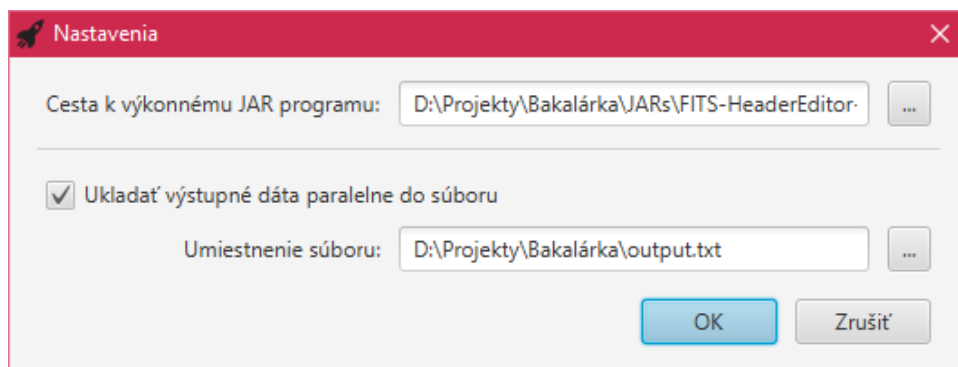
Pri vývoji sa vyskytol problém s načítavaním slovenských a českých reťazcov, ktoré obsahovali znaky s diakritikou. Trieda *ResourceBundle* totiž na načítanie používa východziu znakovú sadu **ISO 8859-1**, ktorá tieto znaky neobsahuje. Bolo teda potrebné vytvoriť pomocnú triedu *UTF8Control*, ktorá už reťazce načítava v správnom kódovaní [13].

Jazyk sa dá prepínať v položke menu **Jazyk** v hornej časti grafickej aplikácie.

5.3.5 Nastavenia programu

V menu **Program** > **Nastavenia** sa nachádza dialóg s nastaveniami aplikácie:

Prvé nastavenie sa týka cesty k JAR súboru s výkonným programom. To znamená, že aby aplikácia fungovala korektne, je potrebné pred spustením túto cestu špecifikovať. Vo východzom nastavení hľadá program súbor **engine.jar** v rovnakom priečinku, z ktorého je spúšaný. Ak ho nenájde, program zobrazí chybový dialóg.



Obr. 5.5: Dialóg s nastaveniami grafickej aplikácie

Výhodou tohto riešenia je, že program nie je závislý na konkrétnom súbore alebo verzii výkonného programu a nemusí sa rekompilovať v prípade, že sa v ňom niečo zmení. Takisto je možné program zameniť za iný s rovnakou funkcionalitou, ale iným spôsobom internej implementácie.

Druhá možnosť sa týka výpisu správ z výkonnej časti paralelne do súboru. Štandardne sa výpis zobrazuje len do časti s textovým výstupom, týmto nastavením je možné ukladať ho paralelne aj do súboru. Je ale potrebné špecifikovať cestu a názov súboru, do ktorého sa bude výpis ukladať.

Toto nastavenie úzko súvisí so súborom *fits.properties*, ktorý výkonný program potrebuje pre svoje fungovanie. Výhodou je, že v prípade, ak sa program spúšťa z grafickej aplikácie, tento súbor nemusí existovať. Aplikácia si ho aj podľa tohto nastavenia sama vygeneruje a po ukončení automaticky zmaže.

Všetky nastavenia (vrátane jazyka) sa ukladajú v systéme a pretrvávajú aj po ukončení programu. Po opätovnom spustení sa znova načítajú, nie je teda potrebné nastavovať aplikáciu po každom jej spustení.

Jadrom dialógu je riadiace trieda *UserPreferencesController* s príslušným FXML súborom, nastavenia sa ukladajú do objektu triedy *Preferences* a s načítavaním a ukladaním nastavení pomáha trieda *PreferencesService*.

6 Súvisiace práce

Na čítanie a úpravu FITS súborov existuje veľké množstvo dostupných nástrojov a programov. Líšia sa množstvom funkcií či zameraním na konkrétne operácie s FITS súbormi. Niektoré nástroje sú však už zastarané a ďalej nevyvíjané, prípadne spustiteľné iba vo vybraných operačných systémoch. Nasledujúci prehľad vyberá iba tie najrelevantnejšie.

Prvým a zároveň jediným nástrojom, ktorý umožňuje hromadnú úpravu FITS hlavičiek, je program **FITS4Win2 Header Data Utility**, konkrétne jedna jeho časť, balíček **F4W2HDU** [14]. Celý program má proprietárnu licenciu a pre dlhšie používanie je potrebné si ho zakúpiť, uvedený balíček je ale dostupný zdarma.

Nástroj má prepracované grafické rozhranie, dokáže načítať viacero FITS súborov a poskytuje náhľad ich hlavičiek. Pomocou začiar-kavacích políčok je možné finálne označiť súbory na úpravu. Vybraté súbory sa však nedajú ďalej filtrovať, napríklad podľa kľúčových slov alebo hodnôt v hlavičke.

Podporuje vkladanie nových záznamov a úpravu či zmazanie existujúcich záznamov. Obsahuje aj malú databázu rôznych kľúčových slov a ich naviazanie na obrázkové vlastnosti, takže pri výbere vlastnosti sa automaticky zobrazí kľúčové slovo, ktoré popisuje jej hodnotu v hlavičke.

Výhodou programu je, že umožňuje pripraviť viacero zmien hlavičkových záznamov a potom ich naraz vykonať. Nevýhodou naopak obmedzenosť iba na platformu MS Windows.

Druhým nástrojom je program **Fv** [15] vyvíjaný pod hlavičkou organizácie NASA HEASARC¹. Umožňuje prehliadať rôzne typy FITS súborov (obrazové, tabuľkové a i.) či už umiestnené lokálne alebo na vzdialenom FTP serveri. Dáta dokáže vyhľadávať a sťahovať aj z katalógov a stránok NASA (napr. *SkyView*, *VizieR*). Ďalej poskytuje pripojenie na platformu **Hera**², ktorá umožňuje rôzne analýzy a pokročilé operácie s FITS súbormi na vzdialenom stroji, takže program slúži iba ako klient na komunikáciu.

1. High Energy Astrophysics Science Archive Research Center

2. Viac informácií na domovskej stránke <http://heasarc.gsfc.nasa.gov/hera/>

Čo sa týka hlavičky, program dokáže pridávať, upravovať a odstraňovať záznamy so všetkých HDU častí súboru. Nedokáže však upravovať viacero súborov naraz, čo je v tomto prípade jeho hlavnou nevýhodou.

Program je aj naďalej vyvíjaný, posledná verzia je zo septembra 2015, no užívateľské rozhranie je veľmi zastarané a neprehľadné. Spusiteľný je na všetkých hlavných operačných systémoch.

V rámci univerzity bola ako súčasť dizertačnej práce vyvíjaná sada skriptov v jazyku Python – **MUNI-FITS-Utils** [16]. Skripty sa skladajú z dvoch častí, prvá hromadne upraví hlavičky FITS súborov do požadovaného formátu, druhá ich hromadne premenuje podľa zadanej konvencie. Ich účelom je zjednotiť formát a názov pred uložením do fotometrického archívu.

Autor práce však ich vývoj zanechal a skripty tak zostali nedokončené.

Ďalšie nástroje, skripty a knižnice sú vyvíjané hlavne v rámci NASA HEASARC (napr. *CFITSIO*, *FTOOLS* a i.) a umožňujú prácu s FITS súbormi, žiaden z nich však neobsahuje funkcionality požadované touto prácou.

7 Záver

Cieľom práce bolo navrhnuť a zostaviť program pre Ústav teoretickej fyziky a astrofyziky Prírodovedeckej fakulty Masarykovej univerzity v Brne, ktorý bude umožňovať upraviť hlavičky viacerých FITS súborov naraz.

Výsledkom je dvojica samostatne spustiteľných nástrojov napísaných v jazyku Java, ktoré je možné používať vo všetkých hlavných operačných systémoch (MS Windows, Linux, Mac OS X) bez dodatočnej rekompilácie.

Prvý nástroj predstavuje výkonné jadro, ktoré sa zaoberá samotnými úpravami hlavičiek a je spustiteľný z príkazového riadku. Obsahuje všetky požadované operácie s hlavičkovými záznamami. Na spustenie nepotrebuje žiadne ďalšie závislosti okrem konfiguračného súboru v rovnakom priečinku, kde sa nastaví jeho výstup. Na čítanie a zapisovanie do FITS súborov využíva voľne prístupnú externú knižnicu.

Druhú časť tvorí grafické rozhranie napísané jazyku JavaFX (nástavba jazyka Java) a poskytuje možnosť pohodlného výberu súborov, nastavenie operácií pomocou myši a prehľadný výpis informácií o priebehu operácie. Súbory je možné filtrovať podľa hlavičkových záznamov a na úpravu vyberať iba niektoré z nich. Rozhranie neposkytuje výkonné jadro na úpravu hlavičiek, volá na to prvú časť nástroja.

Obe časti boli počas vývoja testované a konzultované cieľovou skupinou užívateľov, ktorá poskytla cenné rady a upresnenia požiadavkov. Program tak mohol byť vyvinutý presne podľa ich potrieb.

Textová časť práce obsahuje okrem popisu implementácie a fungovania nástroja aj stručný prehľad FITS formátu a použitých technológií.

Praktická časť sa skladá z dvoch samostatne spustiteľných JAR súborov: jeden s výkonnou, druhý z grafickou časťou.

Zdrojové kódy nástroja, ich dokumentácia a popis sú umiestnené vo verejnom repozitári na serveri GitHub¹, nie je teda problém na túto prácu ďalej nadviazať a rozvíjať ju podľa potrieb.

Medzi možnosti budúceho rozšírenia je možné uviesť podporu pre ďalšie jednoduché alebo pokročilé operácie s hlavičkami, úpravu

1. <https://github.com/supermartzin/fits-header-editor>

hlavičiek tabuľkových FITS súborov, či možnosť náhľadu hlavičiek jednotlivých súborov v grafickom rozhraní.

Nástroj bude ďalej otestovaný zadávateľom práce a v prípade, že s ním bude spokojný, aj nasadený do používania.

Literatúra

- [1] WELLS, D. C.; GREISEN, E. W.; HARTEN, R. H. FITS : A Flexible Image Transport System. *Astronomy and Astrophysics Supplement Series* [online]. 1981, roč. 44, [cit. 2015-12-01], s. 363–370. Dostupný z WWW: <http://adsabs.harvard.edu/abs/1981A&AS...44..363W>.
- [2] PENCE, Dr. William D. (ed.). *A Primer on the FITS Data Format* [online]. 2014 [cit. 2015-12-01]. Dostupný z WWW: http://fits.gsfc.nasa.gov/fits_primer.html.
- [3] NOST. *Definition of the Flexible Image Transport System (FITS)* [online]. 1993 [cit. 2015-12-01]. 88 s. Dostupný z WWW: http://fits.gsfc.nasa.gov/standard10/fits_standard10.pdf.
- [4] IAUFWG. *Definition of the Flexible Image Transport System (FITS): The FITS Standard* [online]. 2010 [cit. 2015-12-01]. Dostupný z WWW: http://fits.gsfc.nasa.gov/standard30/fits_standard30aa.pdf.
- [5] LÉNYI, Peter. *Photometric Archive of Astronomical Images* [online]. 2014 [cit. 2015-12-14]. 47 s. Dostupný z WWW: https://is.muni.cz/th/256333/fi_m_a2/256333_-_Peter_Lenyi_-_Master_s_Thesis_-_Photometric_Archive_of_Astronomical_Images_-_Spring_2014.pdf.
- [6] ARMS, C. R.; FLEISCHHAUER, C.; MURRAY, K. *Flexible Image Transport System (FITS), Version 3.0* [online]. 2013 [cit. 2015-12-01]. Dostupný z WWW: <http://www.digitalpreservation.gov/formats/fdd/fdd000317.shtml#identification>.
- [7] THE APACHE SOFTWARE FOUNDATION. *Apache Maven Project* [online]. 2002–2015 [cit. 2015-12-09]. Dostupný z WWW: <https://maven.apache.org/index.html>.
- [8] FOWLER, Martin. *Inversion of Control Containers and the Dependency Injection pattern* [online]. 2004 [cit. 2015-12-09]. Dostupný z WWW: <http://www.martinfowler.com/articles/injection.html>.
- [9] PENCE, Dr. William D. (ed.). *FITS I/O Libraries* [online]. 2015 [cit. 2015-12-09]. Dostupný z WWW: http://fits.gsfc.nasa.gov/fits_libraries.html.
- [10] LÉNYI, Peter. *Web fotometrického archívu – Manuál k normalizácii* [online]. 2013 [cit. 2015-12-14]. Dostupný po prihlásení z WWW: <https://wasp.cerit-sc.cz/paw/util/Help/norm>.

-
- [11] MICROSOFT CORP. *Command prompt (Cmd.exe) – command-line string limitation* [online]. 2015 [cit. 2015-12-13]. Dostupný z WWW: <https://support.microsoft.com/en-us/kb/830473>.
- [12] BRUTON, Dan. *Heliocentric Julian Day* [online]. [cit. 2015-12-13]. Dostupný z WWW: <http://www.physics.sfasu.edu/astro/javascript/hjd.html>.
- [13] SCHOLTZ, Bauke. *How to use UTF-8 in resource properties with ResourceBundle* [online]. 2015 [cit. 2015-12-20]. Dostupný z WWW: <http://stackoverflow.com/a/4660195/2212098>.
- [14] ASTROSHED. *FITS Header Utility - F4W2HDU (Free)* [online]. 2003–2014 [cit. 2015-12-21]. Dostupný z WWW: <http://astroshed.com/fits4win/f4w2hdudoc.htm>.
- [15] PENCE, William; CHAI, Pan. *Fv: The Interactive FITS File Editor* [online]. 2015 [cit. 2015-12-21]. Dostupný z WWW: <http://heasarc.gsfc.nasa.gov/docs/software/fvtools/fv/>.
- [16] CHRASTINA, Marek; ZEJDA, Miloslav; MIKULÁŠEK, Zdeněk. MUNI-FITS-Utils. In. *Binaries - key to comprehension of the universe*. San Francisco: Astronomical society of the Pacific, 2010, s. 83–84.

A Prílohy

- 1 **FITS Header Editor** – *Používateľský manuál* výkonnej a grafickej časti
- 2 **MUNI CCD Photometric Archive** – *Definícia FITS hlavičky*
(Mgr. Marek Chrastina a RNDr. Miloslav Zejda, Ph.D.)
- 3 spustiteľné programy v podobe JAR súborov
- 4 zdrojové kódy programov