ZÁPADOČESKÁ UNIVERZITA V PLZNI
FAKULTA APLIKOVANÝCH VĚD
KATEDRA KYBERNETIKY

VÝVOJ ÚLOH VZDÁLENÉ A VIRTUÁLNÍ LABORATOŘE ZALOŽENÝCH NA TECHNOLOGII HTML 5

Diplomová práce

Plzeň, 2013 Bc. Jan Skyba

Prohlášení

Předkládám tímto k posouzení a obhajobě diplomovou práci zpracovanou na závěr studia na Fakultě aplikovaných věd Západočeské univerzity v Plzni.

Prohlašuji, že jsem diplomovou práci vypracoval samostatně a výhradně s použitím odborné literatury a pramenů, jejichž úplný seznam je její součástí.

V Plzni dne .....

Poděkování

Na tomto místě bych rád poděkoval Ing. Pavlu Baldovi Ph.D. za odborné vedení, konzultace a připomínky, které mi poskytoval během psaní této práce. Rovněž děkuji Ing. Miroslavu Kocánkovi za poskytnutí části svého kódu, stejně tak jako Ing. Ondřeji Severovi a Ing. Martinu Čechovi za vynikající přístup během konzultací.

Zvláštní poděkování patří mým rodičům za podporu a zázemí během mého studia.

Abstrakt

Tato diplomová práce se zabývá tvorbou vzdálených a virtuálních laboratoří za použití internetové technologie HTML5. V práci je prezentován budoucí standard HTML5, programovací jazyk JavaScript, vektorový grafický formát SVG, řídicí systém REX a tzv. webové sokety. Pomocí těchto nástrojů byly v rámci této práce vyvinuty komponenty, z nichž lze sestavit množství vzdálených a virtuálních laboratoří. Stěžejní část práce je věnována popisu funkcí a možností jednotlivých komponent. Základní z nich je grafický objekt ChartPanel, který slouží k vykreslování obecných (třída AxesPanel) i časových (třída Trend) grafů. Dalšími vyvinutými grafickými objekty jsou Slider (posuvník), BarGraf (sloupcový graf) a Display (displej). Pro praktickou ukázku výše zmíněných grafických objektů byly vytvořeny dvě virtuální laboratoře. První z nich demonstruje možnosti nastavení PID regulátoru s momentovým autotunerem a druhá se zabývá tzv. Smithovým prediktorem. Vyvinuté komponenty jsou dostupné v knihovnách vizualizačních nástrojů REX.UI a REX.UI.CHARTS.

Klíčová slova

Virtuální laboratoř, HTML5, JavaScript, SVG, webové sokety, REX, HMI, trend, posuvník, sloupcový graf, displej

Abstract

This master thesis concerns a creation of remote and virtual laboratories using HTML5 web technology. There are presented several computing technologies such as future standard HTML5, JavaScript programming language, SVG vector graphics format, REX control system and websockets. The new user interface components have been developed using these tools. A variety of remote and virtual laboratories can be built from these components. The main part of this work is dedicated to the description of functions and options of these components. The most complex component of the implemented graphical objects is the ChartPanel, which contains classes for drawing general graphs (AxesPanel) and time graphs (Trend). Other graphical objects developed in this work are Slider BarGraph and Display. These objects are used in the final part of this thesis in order to create two virtual laboratories. The first laboratory demonstrates options of the PID controller with torque autotuner. The second laboratory deals with Smith predictor. Developed components are included into REX.UI and REX.UI.CHARTS visualization libraries.

Key words

Virtual laboratory, HTML5, JavaScript, SVG, websockets, REX, HMI, trend, slider, bargraph, display

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod

Cílem této práce je tvorba vzdálených a virtuálních laboratoří založených na internetové technologii HTML5, respektive postupný vývoj jednotlivých komponent, které následně mohou být použity pro vytvoření množství vzdálených a virtuálních laboratoří. Základním vytvořeným prvkem je časový graf (*trend*) neboli grafický objekt zobrazující hodnoty vstupních signálů a poskytující uživateli nástroje pro práci se zobrazovanými údaji. Dalšími vyvinutými objekty jsou posuvník (*slider*), který umožňuje zadávání hodnot pomocí myši, sloupcový graf (*bargraf*) a displej.

Důvodem pro zadání výše zmíněných úloh založených na HTML5 jsou nové možnosti, které tento budoucí standard přináší do prostředí webových technologií. Pro tvorbu virtuálních laboratoří ve webovém prostředí bylo dříve nutno používat externí technologie (zásuvné moduly) napsané např. v jazyce Java. Budoucí standard HTML5 umožňuje pomocí jazyka JavaScript, jenž je nativně podporovaný webovými prohlížeči, vytvářet grafické prvky např. ve vektorové grafice a komunikovat se serverem pomocí tzv. webových soketů. Všechny tyto technologie jsou zdarma a volně šiřitelné.

Vzdálené a virtuální laboratoře si získávají stále větší popularitu. Jedním z důvodů tohoto stavu je jejich interaktivita, která je s novějšími a dokonalejšími technologiemi stále vyšší. Tyto laboratoře slouží jak ve školách, kde je může využívat množství studentů současně, tak i v reálném světě (např. pro zaškolení nových pracovníků).

Ve vzdálených laboratořích se pracuje s reálným systémem a experimenty prováděné uživatelem jsou skutečné. Přístup do vzdálené laboratoře je zpravidla uskutečňován přes internet.

Při experimentech, které uživatel provádí ve virtuální laboratoři, se pracuje s virtuální reprezentací reality (matematickými modely), což s sebou nese řadu kladů oproti klasické nebo vzdálené laboratoři. Největší výhodou je cena, která nevzrůstá při nutnosti mnohonásobného opakování daného experimentu. Dále také odpadá i potřeba nákupu či údržby přístrojů nutných pro fungování skutečné laboratoře. Mezi další nesporné výhody virtuálních laboratoří patří variabilita simulačního času, který lze dle potřeby zrychlovat, či zpomalovat. Práce ve virtuální laboratoři je bezpečná a navíc lze simulovat její běh i v podmínkách, kterých bychom ve skutečné laboratoři nemohli dosáhnout.

Úvodní část této práce obsahuje seznámení se s vznikajícím standardem HTML5, programovacím jazykem JavaScript, vektorovou grafikou SVG a řídicím systémem REX. Další kapitola se zabývá analýzou požadavků na vizualizaci v HTML5. Největší část této práce je věnována vývoji grafického objektu ChartPanel, který obsahuje třídy pro zobrazování obecných i časových křivek. Další kapitola se zabývá vývojem grafických objektů Slider (posuvník), BarGraf (sloupcový graf) a Display (displej). Vyvinuté grafické objekty položí základ vytvářeným knihovnám vizualizačních nástrojů REX.UI a REX.UI.CHARTS. V poslední kapitole jsou za pomoci výše zmíněných grafických objektů sestaveny dvě virtuální laboratoře. První z nich demonstruje funkce PID regulátoru s momentovým autotunerem, druhá poté možnosti tzv. Smithova prediktoru.

2 Použité technologie

2.1 Internetová technologie HTML5

HTML (*HyperText Markup Language*) [5] je značkovací jazyk poskytující nástroje pro tvorbu webových stránek a zveřejňování publikací na internetu. Jazyk HTML je charakterizován množinou značek (nazývají se tagy) a jejich atributů. Mezi tagy je uzavřen text a tím je určen jeho význam neboli sémantika. Jazyk HTML byl vytvořen v roce 1990 spolu s protokolem HTTP (*HyperText Transfer Protocol*) [7], který umožnil jeho přenos v počítačové síti. V průběhu let jazyk HTML prošel postupným vývojem a standardizacemi. Poslední standardizovaná verze je 4.01 z roku 1999.

Budoucí standard HTML5 je od roku 2007 stále ve stádiu vývoje (*draft*). HTML5 je reakcí na dnešní zastaralost předchozí verze. Na vývoji HTML5 se podílí konsorcium W3C (*World Wide Web Consortium*) a pracovní skupina WHATWG (*Web Hypertext Application Technology Working Group*), přičemž se řídí následující filozofií:

- Nově implementované vlastnosti by měly být založeny na technologiích HTML, CSS (*Cascading Style Sheets*) [6], DOM (*Document Object Model*) [8] a JavaScript. Z toho vyplývá, že HTML5 nelze již považovat pouze za značkovací jazyk, jakým původně HTML byl, ale za celý balíček výše zmíněných technologií.
- Redukce potřeby externích zásuvných modulů (*plugins*), mezi které patří například Flash.
- HTML5 by mělo být nezávislé na zařízení

Z uvedené filozofie vyplývá, že specifikace HTML5 s sebou nutně přináší množství nových vlastností, tagů a sémantiky. Mezi nejzajímavější novinky, jejichž podporu do prostředí webového prohlížeče vznikající standard zavádí, patří například:

- Tag <canvas> pro 2D kreslení.
- Tagy <video> a <audio> pro multimediální obsah.
- Podpora vektorové grafiky.
- Podpora tzv. webových soketů.
- Implementace metody „táhni“ a „pusť“ (*drag and drop*).
- Offline webové aplikace.
- Nové atributy pro webové formuláře.
- Geolokace – získání zeměpisné polohy uživatele.
- Podpora technologie CSS3.

- Lokální úložiště ve webovém prohlížeči.

Zabývat se všemi novinkami, jež specifikace HTML5 přináší do prostředí webových technologií, přesahuje rozsah této práce. Pro účely vytvoření nástrojů využitých pro virtuální laboratoř byla z vyjmenovaných novinek využita zejména podpora vektorové grafiky a webových soketů.

Přestože je specifikace HTML5 stále ve vývoji, již je podporována většinou dnešních prohlížečů (i když ne vždy v plném rozsahu), které ve svých aktualizacích reagují na její nově zaváděné vlastnosti a nástroje.

2.2 Programovací jazyk JavaScript

2.2.1 Historie a vývoj

JavaScript [1] je dynamický multiplatformní programovací jazyk, jenž byl na svém začátku zamýšlen jako jednoduchý nástroj pro doplnění dynamických prvků webové stránky, validaci formulářů atd. Program vytvořený v jazyce JavaScript je spouštěn na straně klienta až po stažení webové stránky (na rozdíl např. od jazyků PHP a ASP běžících na straně serveru). JavaScript je dnes nejpoužívanějším programovacím jazykem na světě.

Autorem jazyka JavaScript je Brendan Eich, který jej vyvinul v roce 1995 ve společnosti Netscape. Šlo o vůbec první dynamický programovací jazyk podporovaný ve webovém prohlížeči (Netscape Navigator 2). Původně se jmenoval Mocha, poté LiveScript. Přejmenování na JavaScript mělo zdůraznit podobnost s jazykem Java. Kvůli tomuto kroku ovšem často dochází k mylnému domnění, že JavaScript je rozšířením jazyka Java pro webový prohlížeč. Skutečnost je taková, že se jedná o odlišné technologie, které mají pouze podobnou syntaxi. Historie tohoto jazyka je poněkud komplikovanější z důvodu, že i společnost Miscrosoft po roce 1995 začala vyvíjet svoji verzi jazyka JavaScript pod názvem JScript. Ke společné standardizaci došlo v roce 1997 asociací ECMA (*European Computer Manufacturers Association*). Dále pak v roce 1998 organizací ISO (*International Organization for Standardization*). Standardizovaná verze se nazývá ECMAScript.

V průběhu let se výrazně změnil význam jazyka JavaScript. Původně zamýšlenému jednoduchému nástroji bylo postupně prohlížeči umožněno přistupovat k objektovému modelu webové stránky (DOM), což umožnilo například měnit či mazat její obsah. Dalším krokem byl nástup technologie AJAX (*Asynchronous JavaScript and XML*) [13], která zpřístupnila jazyku JavaScript možnost komunikace se serverem. Dále docházelo k nástupu knihoven typu Prototype a dalších, které zjednodušují práci s DOM. Nelze opomenout ani standard HTML5, jenž umožňuje pomocí jazyka JavaScript použít ve webových stránkách např. vektorovou grafiku, komunikaci pomocí webových soketů atd.

2.2.2 Rysy objektově orientovaného jazyka

Stejně jako je tomu v případě novinek týkajících se specifikace HTML5, zabývat se všemi aspekty programování v jazyce JavaScript by přesahovalo rozsah této práce. Proto zde nebude rozváděna problematika přístupu jazyka JavaScript k proměnným, polím, funkcím atd. Tato práce je se svým přístupem k jazyku JavaScript založená na postupech představených v [2] a jako vhodný učební nástroj posloužila internetová příručka dostupná na [12].

Při práci na aplikacích popsaných v následujících kapitolách je nutno na JavaScript pohlížet jako na objektově orientovaný programovací jazyk. Tato oblast je poměrně sporná, proto je rozumné se na tento aspekt podívat blíže. JavaScript nepodporuje klasický koncept třídy a její instance, který je známý z ostatních objektově orientovaných programovacích jazyků. Tento mechanismus lze ovšem nahradit tzv. „prototypováním“ a „simulovat“ tak vlastnosti objektově orientovaného programovacího jazyka, který bude umožňovat např. dědění. Za třídu tedy budeme považovat konstrukční funkci neboli konstruktor, který využívá vlastnosti *prototype*. Existuje mnoho technik tvorby tříd v jazyce JavaScript uveřejněných například v [2]. Implementace techniky využité v aplikacích vytvořených v rámci této práce je popsána v kapitole 4.3.

2.3 Vektorová grafika SVG

SVG (*Scalable Vector Graphics*) [10] je grafický vektorový formát popsaný v jazyce XML (*Extensible Markup Language*) [9]. Formát SVG byl navržen pro užití vektorové grafiky ve webovém prohlížeči (ale nejen v něm), která postupně nahrazuje používání rastrové grafiky z důvodů, jakými jsou například jednoduchá změna velikosti a stejná ostrost při libovolném přiblížení. Specifikace SVG poskytuje následující funkčnost:

- Tvorba základních geometrických obrazců, jakými jsou kružnice, elipsa, obdélník a lomená čára (viz tabulku 1).
- Tvorba grafiky pomocí křivek (*paths*). Tato možnost je obecnější a umožňuje vytvářet prakticky jakékoliv grafické obrazce.
- Možnost vytváření elementů typu text.
- Možnost kumulace více elementů do grup.
- Interaktivita – u výše zmíněných grafických prvků lze nastavovat kromě barvy a velikosti, také například jejich průhlednost, vytvářet animace atd.
- Podpora skriptování.

V aplikacích, popsaných v této práci, je využito toho, že SVG element je uložen ve formě stromu, jenž je popsán v XML. Z toho vyplývá, že k SVG lze přistupovat přes DOM a měnit parametry jednotlivých uzlů stromu. K DOM je přistupováno pomocí jazyka

JavaScript. Při tvorbě SVG prvků posloužily informace získané v [11] jako vhodný výukový materiál.

Tvar	Značka
Obdélník	<rect>
Kruh	<circle>
Elipsa	<ellipse>
Úsečka	<line>
Lomená čára	<polyline>
Mnohoúhelník	<polygon>
Křivka	<path>

Tabulka 1: Geometrické tvary v SVG

2.4 Řídicí systém REX

2.4.1 Popis systému

Řídicí systém REX je nástroj určený pro realizaci algoritmů automatického řízení založený na systému funkčních bloků. Jedná se o vyspělou technologii, jež kromě oblasti automatizace a regulace poskytuje i nástroje pro realizaci pokročilých algoritmů automatického řízení.

Teorie automatického řízení se dynamicky rozvíjí, řídicí systém REX tak mimo samotného řízení poskytuje i možnost simulace před uvedením daného zařízení do praxe. REX je kompatibilní s globálně rozšířeným produktem Matlab-Simulink, jehož funkční bloky ovšem nejsou vhodné pro využití v praxi, jelikož jsou založeny pouze na učebnicových algoritmech. REX zavádí vlastní knihovnu funkčních bloků RexLib, které obsahují všechny potřebné funkce pro skutečnou průmyslovou realizaci. Rovněž umožňuje přenesení na cílovou platformu v jazyce C a C++.

Vývojové prostředí pro návrh funkčních schémat poskytuje program RexDraw, funguje podobně jako editor v Matlab-Simulink. Vytvořený soubor ve formátu *.mdl* je přeložen překladačem RexComp do binárního konfiguračního souboru *.rex*. V tomto formátu lze soubor zaslat na konkrétní cílové zařízení a zahájit řízení v reálném čase bez nutnosti odstavení zařízení, tento přenos zajišťuje diagnostický protokol založený na standardu TCP/IP. Na sledování dění při běhu procesu má REX implementovaný diagnostický nástroj RexView.

Cílem této práce není realizace algoritmů řízení pomocí funkčních bloků řídicího systému REX. Tato práce bude nicméně pro účely vizualizace a virtuální laboratoře využívat schémata vytvořená právě v systému REX.

2.4.2 Funkční bloky

Podobně jako v programu Matlab – Simulink pracujeme v systému REX pomocí systému funkčních bloků obsažených v knihovně RexLib. Jednotlivé bloky jsou na základě své funkce dále organizovány do podknihoven (viz tabulku 2). Více o ŘS REX lze nalézt v [4].

EXEC	Konfigurace exekutivy reálného času – konfigurace struktury a časování objektů v reálném čase. Pro použití v programu Simulink jsou zde jen bloky LPBRK a SLEEP.
INOUT	Bloky vstupů a výstupů systému REX – zprostředkování vazby mezi řídícími úlohami a vstupně-výstupními ovladači. Určené většinou pouze pro REX.
MATH	Matematické bloky – Obdoba bloků z programu Simulink určených pro základní matematické operace.
ANAL	Zpracování analogových signálů – do této kategorie patří derivátor, integrátor, dopravní zpoždění apod.
GEN	Generátory signálů – bloky generující testovací, analogové a logické signály.
REG	Bloky pro regulaci – Nejvýznamnější část knihovny RexLib. Obsahuje zejména průmyslové regulátory (P, PI, PID a jejich různé obměny).
LOGIC	Logické řízení – bloky pro kombinační i sekvenční logické řízení, obsahuje jak základní logické operace jako negace a logický součin, tak blok reprezentující sekvenční logický automat ATMT.
ARC	Archivace dat.
SPEC	Speciální bloky.
MC	Bloky Motion Control – bloky zabývající se komplexním řešením pohybu v řídicích úlohách.

Tabulka 2: Knihovny funkčních bloků řídicího systému REX

2.5 Komunikační rozhraní na bázi webových soketů

Webové sokety (*WebSockets*) [16] jsou definovány jako součást připravovaného standardu HTML5. Obecně lze říci, že soket je popsán dvojicí IP adres, protokolem a portem, který je použit pro komunikační spojení mezi dvěma procesy. Rozhraní webových soketů zavedené do jazyka JavaScript definuje duplexní spojení typu plný duplex (*full-duplex*) pro komunikaci klient – server. Spojení typu plný duplex umožňuje současnou obousměrnou komunikaci. Ta je v prostředí webových technologií velmi komplikovanou záležitostí. Webové sokety tuto problematiku zjednodušují.

Technologie obousměrné komunikace typu klient – server je klíčová pro možnost vytváření webových aplikací, které budou fungovat v reálném čase (*real-time*). Využití této komunikace je tudíž nezbytné pro provoz virtuálních webových laboratoří, jejichž tvorbou se tato práce zabývá. Před definováním webových soketů byla komunikace klient – server omezena např. na způsob, kdy se klient dotazoval v pravidelných intervalech serveru, zda nemá nová data (technika „heartbeat“).

Technologie webových soketů vychází ze starších technologií. Technologie AJAX umožňovala zadat jednorázový asynchronní požadavek od klienta na server přes HTTP protokol. Technologie Comet [15] zařizovala dlouhodobé HTTP spojení. Technologie webových soketů automaticky nastavuje komunikační tunel pomocí HTTP příkazu CONNECT, ten otevře TCP/IP spojení na určeném portu a umožní přenos dat. Výhodou webových soketů je jednoduchost jejich implementace v jazyce JavaScript jako třída WebSocket. Její instance se např. vytvoří:

```
var mujWebSoket = new WebSocket("ws://adresa_serveru/");
```

Vytvořená instance třídy WebSocket poskytuje události typu *onopen* pro otevření komunikace, *onclose* pro uzavření komunikace a *onmessage* pro samotnou výměnu dat. Zprávy přicházejí a odcházejí v podobě řetězců. Událost *onmessage* je volána vždy, když ze serveru přijdou data. V aplikacích virtuální laboratoře popsaných v této práci je využíván server webových soketů řídicího systému REX.

3 Analýza požadavků na vizualizaci v HTML5

3.1 SCADA/HMI

SCADA neboli *Supervisory Control And Data Acquisition* je systém shromažďující v reálném čase data z čidel, programovatelných automatů (PLC), vzdálených modulů (RTUs) atd., která posílá na další zpracování do centrálního počítače, jenž generuje řízení. Součástí systému SCADA jsou obvykle vstupně-výstupní hardware, regulátory, HMI, komunikace, databáze a software. Systémy SCADA se v praxi používají již od 60. let 20. století zejména v rozsáhlých distribuovaných systémech (výroba elektřiny, chemický průmysl). Systémy obsahují i několik tisíc vstupů a výstupů.

HMI neboli *Human Machine Interface* je vizualizační operátorské rozhraní, které zobrazuje dispečerovi data z měření a zároveň mu umožňuje zadávat příkazy. Obvykle mluvíme o grafickém prostředí (GUI), které je složeno z jednotlivých grafických objektů. Ty mohou být buď statické, nebo dynamické. Statické grafické objekty v průběhu vizualizace nijak nemění své vlastnosti. Takovým statickým objektem je například bitmapa nebo text. Dynamické grafické objekty jsou propojeny s hodnotou sledovaného výrazu. Změna této hodnoty způsobí změnu některé vlastnosti objektu. Mezi dynamické objekty řadíme například displej (objekt pro zobrazení číselné hodnoty), textbox (objekt pro zadávání hodnoty), button (tlačítko), bargraf (sloupcový graf), slider (posuvník), spínač, ukazatel, alarm a trend (objekt zobrazující průběhy měřených veličin).

SCADA/HMI systémy jsou v dnešní době vyvíjeny řadou českých i zahraničních společností.

3.2 Požadavky na vizualizaci v HTML5

Požadavkem na tuto práci bylo přenést některé z funkcí SCADA/HMI systémů do prostředí webového prohlížeče. HTML5 definuje nástroje, které umožňují použít ve webovém prohlížeči grafiku. V této práci je konkrétně využita možnost implementace grafického vektorového formátu SVG, který je popsaný v jazyce XML a poskytuje možnost skriptování jazykem JavaScript. Hlavními výhodami takové vizualizace jsou jednoduchost přístupu k rozhraní přes webový prohlížeč, který je v dnešní době všudypřítomný, možnost vytvořit virtuální laboratoř bez nutnosti instalace zásuvných modulů (JAVA applety apod.) a využití programovacího jazyka JavaScript, který je volně šiřitelný. Specifikace HTML5 je v době psaní této práce stále ve stádiu vývoje, je ovšem pravděpodobné, že většina vizualizačních nástrojů na ní bude v budoucnu založena, proto je potřeba se touto problematikou zabývat.

V rámci této práce byl pro virtuální laboratoř vytvořen zejména grafický objekt ChartPanel, který obsahuje objekty typu AxesPanel (Trend). Jedná se o zdaleka nejsložitější prvek z výše zmiňovaných grafických objektů a jeho více než půlroční vývoj představuje přibližně dvě třetiny této diplomové práce. V následující kapitole budou

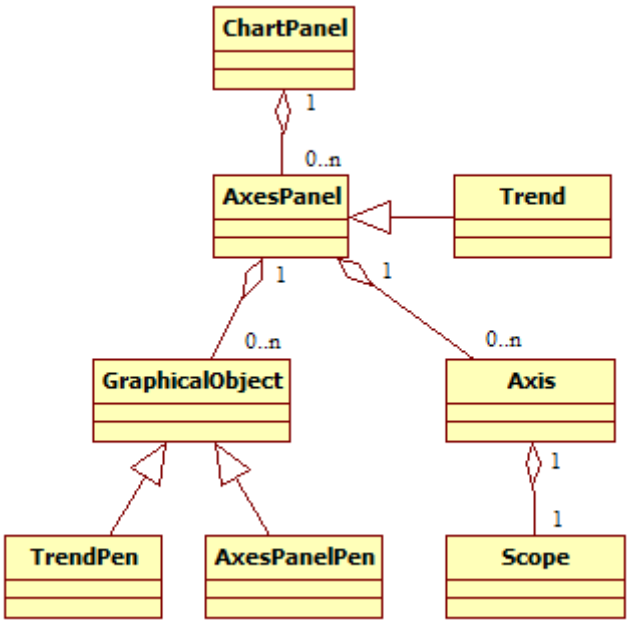
podrobně popsány všechny vrstvy a funkce tohoto objektu, stejně tak bude vysvětlen způsob jeho použití. Dalšími implementovanými objekty, popsanými v následujících kapitolách, jsou bargraf (sloupcový graf), slider (posuvník) a displej.

4 Grafický objekt ChartPanel

Základní myšlenkou, vedoucí k zadání práce na grafickém objektu ChartPanel napsaném v jazyce JavaScript, bylo vytvořit vizualizační nástroj s podobnou funkčností, jakou má například aplikace Rex Trend napsaná v jazyce Java, který ale bude fungovat přímo ve webovém prohlížeči jako nástroj pro vykreslování obecného i časového grafu. Tento způsob je efektivnější, než využití nástroje napsaného např. ve výše zmíněném jazyce Java, pro jehož použití v prostředí webového prohlížeče je nutná instalace dodatečných zásuvných modulů. Před zahájením prací na specifikaci HTML5 bylo podobné řešení nemyslitelné. Grafický objekt ChartPanel se stane základní komponentou vytvářené knihovny vizualizačních nástrojů REX.UI.CHARTS.

4.1 Architektura grafického objektu ChartPanel

Po úvaze byla zvolena následující architektura. Hlavní třída typu ChartPanel obsahuje jednotlivé třídy typu AxesPanel (Trend). AxesPanel je třída sloužící k vykreslení obecného grafu, třída Trend je od ní zděděna a upravuje některé své vlastnosti, aby mohla sloužit k vykreslování časových křivek. Každá z nich má v hierarchii pod sebou třídy typu Axis reprezentující jejich osy. AxesPanel (Trend) má v hierarchii pod sebou dále třídu GraphicalObject, od které jsou zděděny třídy AxesPanelPen (TrendPen). AxesPanelPen reprezentuje obecné pero grafu a TrendPen představuje pero pro časové křivky. Každá třída Axis má pod sebou třídu Scale, jež obsahuje vlastnosti, které určují to, co třída Axis vykresluje. Pro přehlednost je architektura grafického objektu ChartPanel zobrazena na obrázku 1.



Obrázek 1: Architektura grafického objektu ChartPanel

4.2 Požadavky na grafický objekt ChartPanel

Požadavky na funkčnost grafického objektu ChartPanel jsou následující:

- Podpora webových prohlížečů Chrome a Firefox.
- Komunikace se serverem webových soketů řídicího systému REX.
- Graf pro zobrazení obecných (AxesPanel) i časově závislých (Trend) křivek.
- Možnost vertikálního a horizontálního rozdělení (*split*) objektu AxesPanel (Trend).
- Zobrazení ikon jednotlivých per (AxesPanelPen, TrendPen), možnost vypnout jejich zobrazení a možnost přetahovat pera myší mezi objekty typu AxesPanel (Trend) v rámci jednoho objektu typu ChartPanel.
- Možnost spojení vodorovné i svislé osy v rámci jednoho objektu ChartPanel.
- Možnost zamknutí vodorovné i svislé osy v objektu AxesPanel (Trend).
- AxesPanel (Trend) bude mít tlačítka zpřístupňující následující funkčnost:
    - Přepínání mezi zobrazením bodů a křivek,
    - funkce pro zobrazení detailu (zoom) objektu AxesPanel (Trend),
    - funkce pro zpřístupnění posunu v objektu AxesPanel (Trend) tahem myši,
    - funkce *autoscale* pro zobrazení křivek v maximálním rozsahu.
- Při otáčením kolečka myši nad objektem AxesPanel (Trend) probíhá přibližování či oddalování.

4.3 Terminologie

Třídy grafického objektu ChartPanel jsou součástí vytvářené skupiny vizualizačních nástrojů REX.UI.CHARTS. Výjimku tvoří třída Axis, reprezentující osu grafu, jež se nachází ve skupině REX.UI. Je tomu tak z důvodu, že třída Axis bude využita pro vytvoření osy i v dalších vizualizačních nástrojích. V rámci této práce se jedná o třídu BarGraf, která je zařazena do REX.UI.CHARTS a o třídu Slider, která se nachází ve skupině REX.UI. Dále je využívána skupina REX.HELPERS obsahující pomocné nástroje. Realizace grafického objektu ChartPanel v programu vypadá takto:

```
REX.UI.CHARTS.ChartPanel = function(id, settings){…}
```

Z důvodu, že JavaScript ve své podstatě není objektově orientovaný programovací jazyk a tudíž nemá klasické třídy, jaké můžeme znát například z programovacího jazyka JAVA a dalších, musely být nalezeny způsoby jak tento problém obejít (viz 2.2.2). V této práci je využit způsob vytvoření třídy pomocí využití tzv. konstrukční funkce. Klíčová vlastnost je *prototype*, kterou má každá funkce a jejíž výchozí hodnota je prázdný objekt. Tento způsob umožňuje v jazyce JavaScript dědění.

Návratovou proměnou při deklaraci třídy ChartPanel je objekt that, jež je v kódu realizován takto:

```
var that = object(REX.BASE.Component(id, 'rex_ChartPanel'));.
```

Čili třída ChartPanel je zděděna od obecné pomocné komponenty (třídy) pomocí funkce:

```
function object(o) {
      function F() { }
      F.prototype = o;
      return new F();
}
```

Tato funkce zajistí, že nový objekt je zděděn od objektu o.

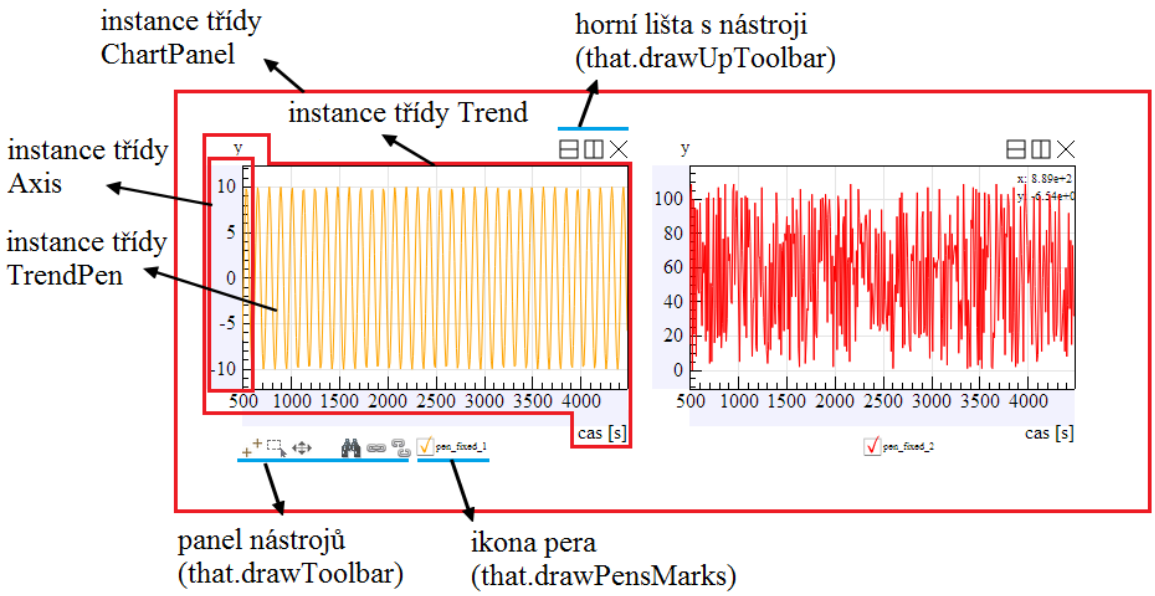
V následujících odstavcích je používán pojem třída, ačkoli toto označení nemusí být považováno za zcela přesné. Tímto způsobem jsou vytvářeny všechny třídy grafického objektu ChartPanel (obrázek 1). Z důvodu přehlednosti jsou v textu použity zkrácené názvy jednotlivých tříd (bez REX.UI.CHARTS popř. bez REX.UI), což je možno pozorovat již na obrázku 1. U každé třídy bude uvedena tabulka s přehledem jejích funkcí, které budou popsány, popř. na příkladech podrobněji rozebrány, pro lepší pochopení jejich účelu. Funkce s předponou that jsou veřejné, jelikož jsou součástí výše zmíněného návratového objektu that.

4.4 Třída ChartPanel

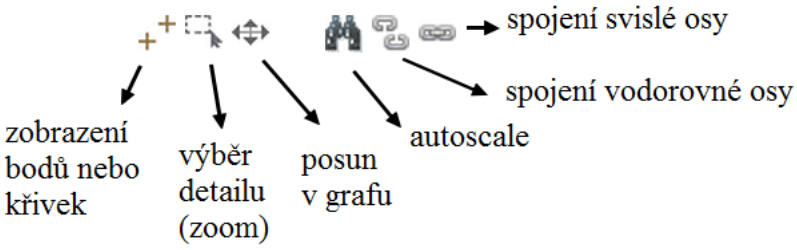
Jak bylo uvedeno výše, ChartPanel je hlavní třída, do níž jsou vkládány jednotlivé instance třídy AxesPanel pro vykreslení obecných křivek a Trend pro vykreslení časových průběhů. Grafický objekt ChartPanel je pro přehlednost s popsanými vnořenými objekty zobrazen na obrázku 2. Vytvoření instance třídy ChartPanel má následující tvar:

```
nazevChartPanel = REX.UI.CHARTS.ChartPanel(id,settings)
```

V proměnné id je definován název instance třídy ChartPanel a do objektu settings uživatel zadává údaje, dle kterých je určeno, např. zda bude vytvořen panel nástrojů popsaný na obrázku 3, který třída ChartPanel vytváří (viz tabulku 3).



Obrázek 2: Popis jednotlivých složek grafického objektu ChartPanel

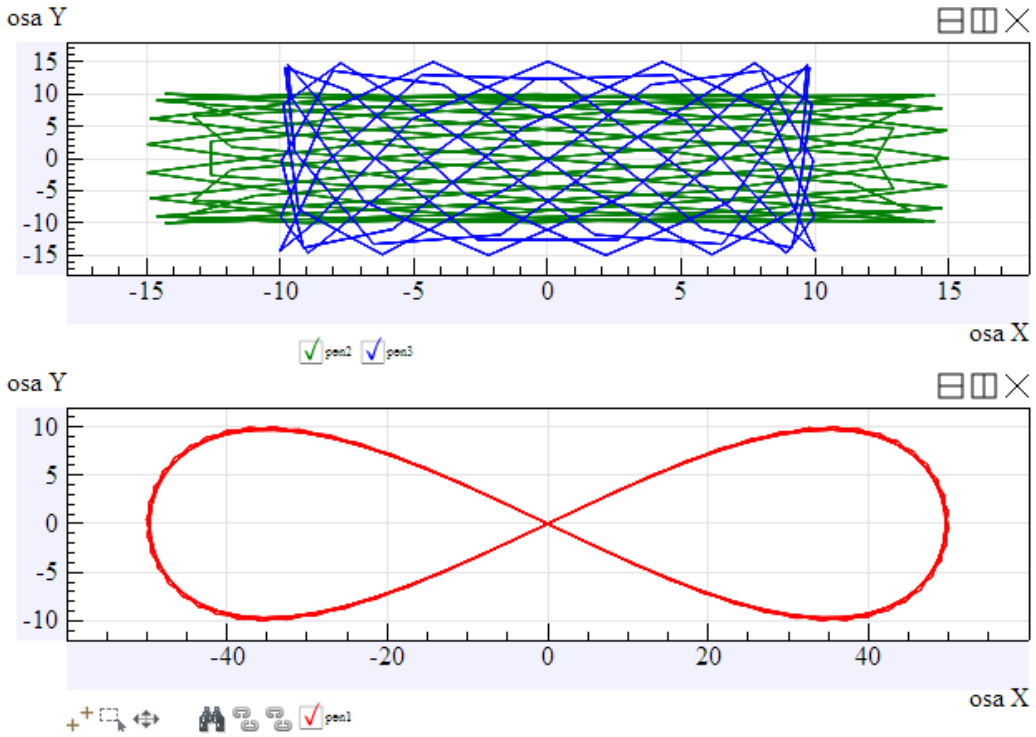


Obrázek 3: Funkce panelu nástrojů

Parametr	Popis a možnosti nastavení
settings.drawToolBar (that.drawToolBar)	Parametr povoluje, či zakazuje vykreslování panelu nástrojů v třídě ChartPanel. Pokud uživatel nastaví hodnotu na *true* je panel nástrojů povolen, při *false* je zakázán. Pokud uživatel tyto hodnoty při vytváření instance třídy ChartPanel nezadá, jsou automaticky nastaveny na hodnotu *true*.
that.drawUpToolBar (that.drawUpToolBar)	Parametr povoluje, či zakazuje vykreslování horní lišty s nástroji nad každým objektem AxesPanel (Trend) v objektu ChartPanel. Pokud uživatel nastaví hodnotu na *true* je vykreslení lišty povoleno, při *false* je zakázáno. Pokud uživatel tyto hodnoty při vytváření instance třídy ChartPanel nezadá, jsou automaticky nastaveny na hodnotu *true*.

Tabulka 3: Nastavitelné parametry objektů settings a that v třídě Chartpanel

Panel nástrojů (obrázek 3) obsahuje ikonu pro přepínání mezi zobrazením bodů a křivek, pro výběr detailu (zoom), pro posun myší v grafu, *autoscale* a ikony pro spojení vodorovné či svislé osy. Panel nástrojů se vytváří na této úrovni, protože je společný pro všechny objekty AxesPanel (Trend) v objektu ChartPanel. Dále se zde generují ikony per (obrázek 2), které jsou umístěny pod objekty AxesPanel (Trend), ke kterým náleží. Ty jsou na této úrovni tvořeny z důvodu možnosti přemisťovat tyto ikony tažením myši mezi jednotlivými objekty AxesPanel (Trend) a tím měnit umístění křivek neboli instancí tříd AxesPanelPen (TrendPen), které jsou těmito ikonami reprezentovány. Posledním nástrojem generovaným třídou ChartPanel je horní lišta nad každým objektem AxesPanel (Trend). Ta umožňuje horizontální a vertikální rozdělení (*split*) objektu AxesPanel (Trend) na dvě části, dále pak zavření objektu AxesPanel (Trend). Horní lišta je generována na této úrovni z důvodu, že rozdělení nebo zavření objektu AxesPanel (Trend) ovlivní i okolní objekty AxesPanel (Trend), které přizpůsobí své rozměry situaci, která nastane. Na obrázku 4 se nachází grafický objekt ChartPanel obsahující dvě instance třídy typu AxesPanel. pod každým objektem AxesPanel jsou ikony jeho per. Pod objektem ChartPanel se nachází panel nástrojů.



Obrázek 4: Grafický objekt ChartPanel obsahující dvě třídy typu AxesPanel

Pro lepší pochopení funkčností této třídy jsou její funkce podrobněji popsány v tabulce 4.

Funkce	Popis
create()	Definuje SVG grupy třídy ChartPanel, do kterých budou ukládány grafické prvky třídy AxesPanel (Trend), dále pak prvky panelu nástrojů a ikon per.
that.addAxesPanel (axesPanel)	Přidá AxesPanel (Trend) do objektu ChartPanel.
that.updateChartPanel()	Zavolá funkce třídy AxesPanel (Trend), které vykreslí jeho obsah. Dále zavolá funkce třídy ChartPanel pro vykreslení panelu nástrojů a ikon per pod jednotlivými grafy.
that.getAxesPanelByIndex (index)	Vrátí žádaný AxesPanel (Trend) podle zadaného indexu.
that.getAxesPanelById(id)	Vrátí žádaný AxesPanel (Trend) podle id, tedy jeho názvu.
that.penFromAxesPanel (penIconGroup)	Funkce vrátí AxesPanel (Trend) ze kterého je pero, jehož ikona, představovaná SVG elementem typu grupa (*g*), do funkce vstupuje.
that.getChartPanelPens()	Vrátí všechna pera ze všech objektů typu AxesPanel (Trend) v objektu ChartPanel.
that.zoomXJoinedAxes (xFrom,xTo)	Změní rozsah vodorovné osy všech objektů AxesPanel (Trend) v objektu ChartPanel.
that.zoomYJoinedAxes (yFrom,yTo)	Změní rozsah svislé osy ve všech objektech typu AxesPanel (Trend) v objektu ChartPanel.
that.zoomAxesPanels (xFrom,xTo,yFrom,yTo)	Změní rozsah vodorovné i svislé osy ve všech objektech AxesPanel (Trend) nacházejících se v objektu ChartPanel.
that.createNewAxesPanel (type, name, width, height, x, y, margin, xaxis, yaxis)	Dle vstupních parametrů vytvoří nový objekt typu AxesPanel (Trend).
that.drawToolBar()	Pod nejspodnějším objektem typu AxesPanel (Trend) vytvoří panel nástrojů zpřístupňující přepínání mezi zobrazením bodů a křivek, výběr detailu (zoom), posouvání myší v objektu AxesPanel (Trend) a možnost zamknout vodorovnou či svislou (viz obrázek 3).
that.drawUpToolBar()	Vytvoří nad každým objektem AxesPanel (Trend) panel s tlačítky zpřístupňujícími možnost horizontálního a vertikálního rozdělení grafu a zavření celého objektu AxesPanel (Trend). Viz obrázek 2.
that.drawPensMarks()	Pod každým objektem typu AxesPanel (Trend) vykreslí ikony per, které obsahuje. Zpřístupňuje možnost zaškrtnutím okna (*checkbox*) zapnout či vypnout zobrazení pera v grafu. Rovněž při kliknutí na ikonu pera volá funkce zajišťující jeho případný přesun mezi objekty AxesPanel (Trend). Např. na obrázku 2 se pod oběma objekty typu AxesPanel nachází ikony jejich per.
that.movePen (pen, fromAxesPanel, toAxesPanel)	Přesune pero do zvoleného objektu AxesPanel (Trend).
that.handleEvent(evt)	Tato funkce je volána vždy, když nastane událost myši. Na úrovni třídy ChartPanel je použita událost *mousemove* (pohyb myši), která v této třídě konkrétně obstarává vykreslování ikonky pera při přesouvání. Dále pak událost

	mouseup, která nastane, když uživatel pustí tlačítko myši. Ta zajišťuje zavolání příslušných funkcí pro přesun pera z jednoho objektu typu AxesPanel (Trend) do druhého.
selectElement(evt)	Označí příchozí SVG element (využíváno pro přesun per) a zpřístupní jej pro další manipulaci.
dynamicSort(property)	Seřadí pole dle vstupní vlastnosti.
findClosestAxesPanel (panelsAr-ray,closingPanel)	Vrátí příchozímu objektu AxesPanel (Trend) seznam (seřazený od nejbližšího) jeho sousedních objektů typu AxesPanel (Trend) zleva, zprava, zdola i shora.
resizeClosestAxesPanel (closestAP, aP, direction)	Změní velikost objektu AxesPanel (Trend) umístěného v objektu ChartPanel tak, aby zaplnil místo, které zabíral jiný objekt AxesPanel (Trend).
setPenStatus(evt)	Při kliknutí na zaškrtávací okno (*checkbox*) u ikony pera změní jeho status (zaškrtnutý nebo prázdný) a současně zjistí, ke kterému peru v objektu AxesPanel (Trend) náleží a změní jeho viditelnost, aby odpovídala statusu zaškrtávacího okna (*checkbox*).
removeElement(id)	Najde a odstraní element podle jeho id.

Tabulka 4: Přehled funkcí třídy ChartPanel

Samotný výčet a popis funkcí třídy ChartPanel v tabulce 4 nestačí k pochopení důležitých souvislostí. Z tohoto důvodu byly vybrány následující příklady, ve kterých budou funkce třídy ChartPanel použity.

- Rozdělení (*split*) a zavření objektu AxesPanel (Trend):

  Stejným způsobem, jakým se funkce that.handleEvent uvedená v tabulce 4 stará o reakce na události vyvolané myší na úrovni třídy ChartPanel, tak i jednotlivé SVG elementy mohou obsahovat funkce (*eventListener*) zajišťující reakce na události vyvolané nad daným elementem. V případě vytváření horní lišty s panelem nástrojů nad každým objektem AxesPanel (Trend) funkcí that.drawUpToolBar je ikonám symbolizujícím horizontální a vertikální dělení přiděleno jak hlídání události *mouseout*, které způsobí pouze zvýraznění ikony při přejetí myši, tak i hlídání události *mousedown* (stisknutí levého tlačítka myši nad ikonou). Pokud toto nastane, je volána funkce that.changeXYSize (tabulka 6), která patří třídě AxesPanel. Tato funkce změní velikost objektu AxesPanel (Trend) na polovinu. Dále je volána funkce that.createNewAxesPanel, která na zbylé polovině vytvoří nový objekt AxesPanel (Trend).

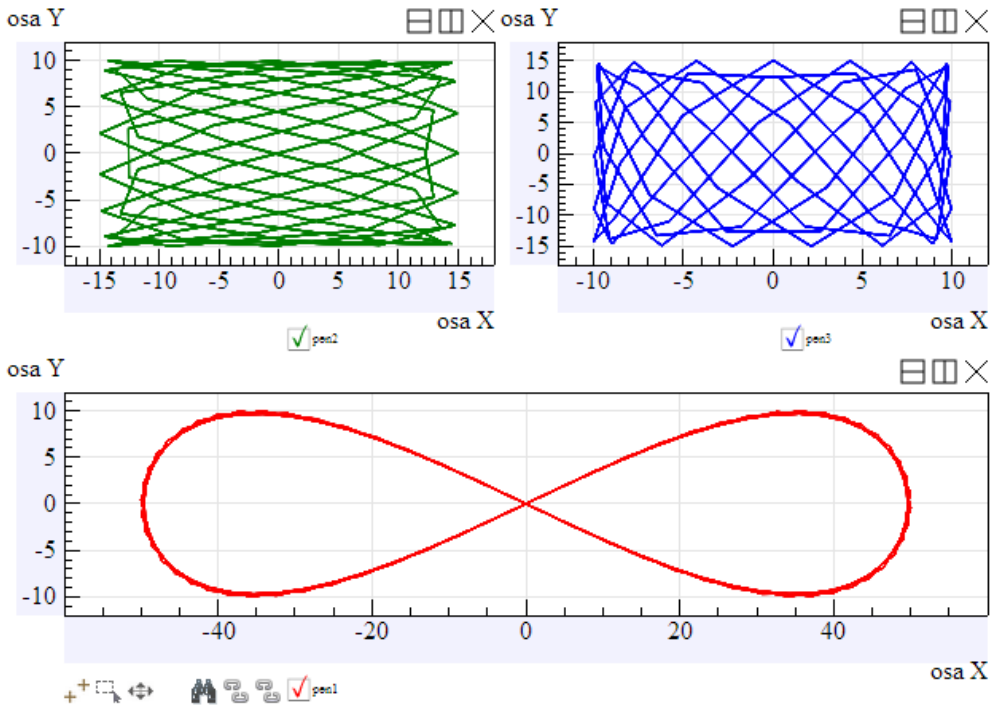
  Podobným způsobem je přiřazeno hlídání událostí i k ikoně pro zavření objektu AxesPanel (Trend). Pokud zde nastane událost *mousedown*, je volána funkce findClosestAxesPanel. Ta vrátí pole nejbližších objektů typu AxesPanel (Trend) a současně i směr, ve kterém se dané objekty nachází. Dále je volána funkce resizeClosestAxesPanel, která pomocí funkce that.changeXYSize změní velikost nejbližšího objektu typu AxesPanel (Trend), tak, aby zaplnila místo po tom, kte-

rý byl zavřen funkcí removeElement. Pera zavřeného objektu AxesPanel (Trend) jsou přesunuta prostřednictvím funkce that.movePen.

- Přesun pera mezi dvěma objekty typu AxesPanel (Trend):

  Tak jako v předchozím bodu se i při vytváření ikon per symbolizujících objekty typu AxesPanelPen (TrendPen), nacházejících se v daném objektu AxesPanel (Trend), k těmto SVG elementům přiřadí hlídání výskytu událostí myši. Událost *mousedown* se u zaškrtávacího pole (*checkbox*) projeví voláním funkce setPenStatus (viz tabulku 4). Pokud nastane událost *mousedown* nad textem, jenž odpovídá názvu pera, je volána funkce selectElement, která uloží matici údajů o poloze ikony. Ta je následně měněna (a tím i pozice ikony) při události *mousemove*, jejíž výskyt je hlídán funkcí that.handleEvent. Stejná funkce zachytí i událost *mouseup*. Při jejím výskytu dojde ke kontrole funkcí that.isInAxesPanel náležící třídě AxesPanel (Trend), která zjistí, nad kterým objektem AxesPanel (Trend) událost nastala. Nakonec je volána funkce that.movePen zajišťující přesun pera.

- Příklad užití předchozích dvou bodů vidíme na obrázku 5, kdy bylo nejprve použito horizontální rozdělení (*split*) horního objektu typu AxesPanel. Poté do něj bylo přesunuto pero s názvem pen3. Vycházelo se ze situace na obrázku 4. Dále je použito volání funkce that.autoScale prostřednictvím panelu nástrojů (obrázek 3). Popis funkce that.autoScale z třídy AxesPanel (Trend) je k dispozici v tabulce 6 a její principy jsou rozebrány v příkladech v 4.5.



Obrázek 5: Rozdělení objektu AxesPanel a přesun pera v rámci objektu ChartPanel

4.5 Třídy AxesPanel a Trend

Třídy AxesPanel a Trend jsou nejdůležitějšími třídami z celkové hierarchie grafického objektu ChartPanel, protože obstarávají vykreslování křivek v grafu a reakce na uživatelem zvolené akce (zobrazení bodů, výběr detailu, posun v grafu tažením myši). Třídy se od sebe liší tím, že AxesPanel je určena pro vykreslení obecné křivky a obsahuje obecná pera typu AxesPanelPen, kdežto třída Trend je určena pro vykreslení časové křivky a obsahuje pera typu TrendPen. Instance těchto tříd vytvoříme následujícím způsobem:

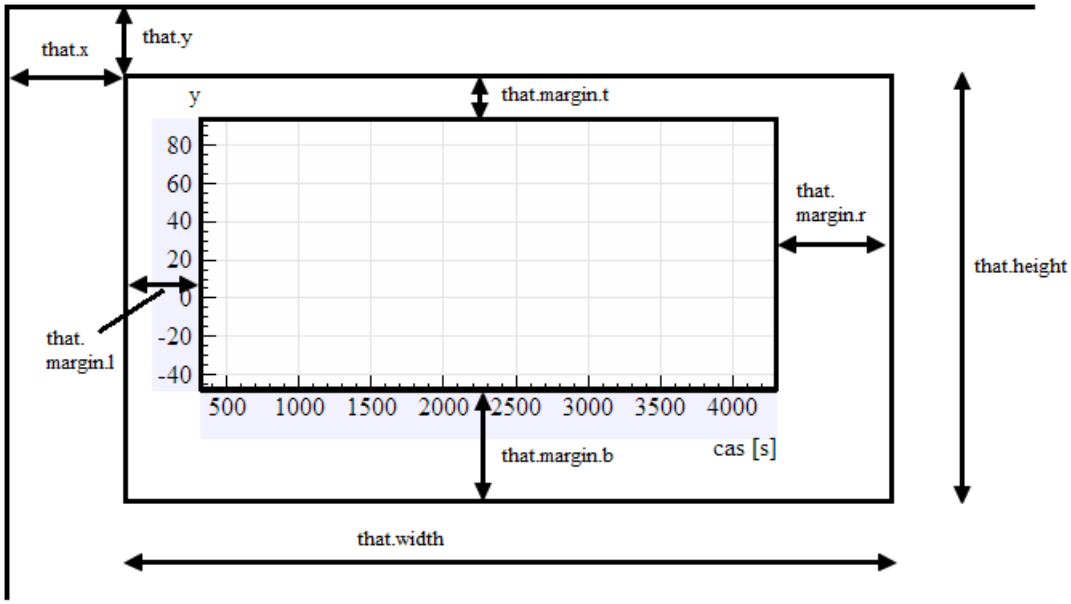
```
nazevAxesPanel = REX.UI.CHARTS.AxesPanel(id,settings)
```

Respektive:

```
nazevTrend = REX.UI.CHARTS.Trend(id,settings)
```

V kódu je implementace taková, že časový Trend je zděděn od třídy AxesPanel a poté mění některé její vlastnosti, jako např. odsazení (*offset*). Z tohoto důvodu je v textu níže psáno pouze o třídě AxesPanel, jelikož zděděný Trend obsahuje stejné funkce.

Při vytváření instance třídy nastavuje uživatel pomocí parametru id její název. Vstupní objekt settings nese důležité údaje o umístění, rozměrech, odsazení, barvách a dalších vlastnostech objektu AxesPanel (viz tabulku 5). Nastavitelné možnosti umístění objektu AxesPanel ve webové stránce jsou znázorněny na obrázku 6.



Obrázek 6: Nastavitelné parametry pozice objektu AxesPanel

Parametr	Popis a možnosti nastavení
settings.x (that.x)	Horizontální souřadnice (v pixelech) horního levého rohu (viz obrázek 6) objektu AxesPanel na webové stránce. Pokud horizontální souřadnice nebude definována při vytváření instance třídy, bude jí přidělena defaultní hodnota.
settings.y (that.y)	Vertikální souřadnice (v pixelech) horního levého rohu objektu AxesPanel na webové stránce (viz obrázek 6). Při neuvedení jí bude přidělena defaultní hodnota.
settings.width (that.widht)	Šířka objektu AxesPanel v pixelech. Při neuvedení bude nastavena defaultní hodnota (viz obrázek 6).
settings.height (that.height)	Výška objektu AxesPanel v pixelech. Při neuvedení bude nastavena defaultní hodnota (viz obrázek 6).
settings.margin (that.margin)	Objekt definující odsazení od okrajů objektu AxesPanel (při nastavení všech hodnot na hodnotu 0 bude graf vykreslen od počátku definovaného hodnotami that.x a that.y a v maximálním rozsahu nastavené šířky a výšky). Nastavují se parametry pro: • Odsazení zleva - l, • odsazení zprava - r, • odsazení shora - t, • odsazení zdola - b. Možnosti nastavení jsou znázorněny na obrázku 6. Při neuvedení bude nastavena defaultní hodnota.
settings.backColor (that.backColor)	Barva pozadí objektu AxesPanel. V případě neuvedení je nastavena na hodnotu *white* (bílá).
settings.backColorIntensity (that.backColorIntensity)	Intenzita barvy pozadí objektu AxesPanel. V případě neuvedení je nastavena na hodnotu 0.2 (20 %).
settings.x_axis	Objekt obsahující nastavení vlastností vodorovné osy (instance třídy Axis). Možnost zadat její vlastnosti je zde, protože třda AxesPanel si při vytváření své instance automaticky nastaví vlastnosti své vodorovné i podélné osy (více v tabulce 9).
settings.y_axis	Analogický objekt k settings.x_axis pro svislou osu.

Tabulka 5: Nastavitelné parametry objektů settings a that třídy AxesPanel

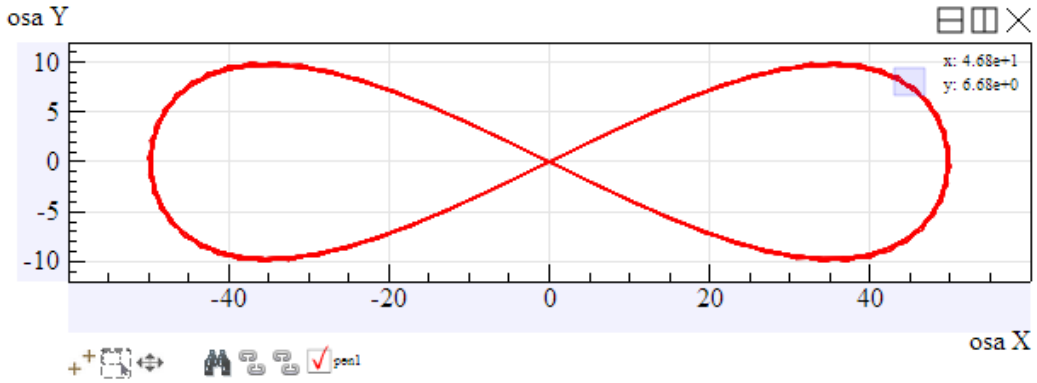
Pro lepší pochopení funkčností této třídy jsou podrobněji popsány její funkce v tabulce 6 a dále jsou na příkladech vysvětleny principy některých základních nástrojů, které poskytuje (výběr detailu, posun v grafu, zobrazení bodů nebo křivek). Jedná se tedy především o prostředky obstarávající funkčnost panelu nástrojů (obrázek 3), který byl vytvořen již třídou ChartPanel.

Funkce	Popis
create()	Vytvoří SVG grupy pro vykreslování křivek a pro uživatelsky aktivní plochy (graf, levý a spodní okraj).
that.redraw(x,y)	Načte a vykreslí AxesPanel a všechny jeho grafické prvky.
that.callParentXZoom (min,max)	Zavolá funkci rodiče (ChartPanel), která zavolá funkce pro výběr detailu (zoom) ve všech objektech typu AxesPanel, které se v něm nachází. Používá se při spojených vodorovných osách.
that.callParentYZoom (min,max)	Viz that.callParentXZoom. Používá se při spojených svislých osách.
that.changeXYSize (x,y,ax,ay)	Změní rozměry objektu AxesPanel.
that.addPen(pen)	Přidá pero do objektu AxesPanel.
that.removePen(pen)	Vyjme pero z objektu AxesPanel.
that.getPenIndex(pen)	Příchozímu peru (objekt AxesPanelPen nebo TrendPen) vrátí index odpovídající jeho umístění v poli per v objektu AxesPanel.
that.clearPen()	Vymaže pole per.
that.getPens()	Vrátí pole per.
that.showPointStatus (status)	Změní status pro zobrazení bodů nebo křivek v objektu AxesPanel.
that.mouseZoomStatus (status)	Změní status umožňující výběr detailu (zoom) v objektu AxesPanel.
that.mouseMoveStatus (status)	Změní status umožňující posun tahem myši v objektu AxesPanel.
that.updateAxesPanel()	Vykreslí všechny křivky (body) v objektu AxesPanel.
that.autoScale()	Zobrazí křivky v grafu (objekt AxesPanel) v jejich maximálním rozsahu.
that.getAxisByName (axis,name)	Vrátí osu (objekt typu Axis) dle její orientace a názvu.
that.getPenByIndex (index)	Vrátí pero (objekt AxesPanelPen nebo TrendPen) z pole per dle indexu odpovídajícímu jeho umístění v poli per.
that.getPenByName (name)	Vrátí pero (objekt AxesPanelPen nebo TrendPen) z pole per dle jeho jména.
that.setBackgroundColor (color,intensity)	Změní barvu pozadí v dané intenzitě.
that.setParent(parent)	Nastaví jako rodiče příslušný ChartPanel.
that.setJoinXAxis(status)	Nastaví parametr pro spojení vodorovné osy.
that.setJoinYAxis(status)	Nastaví parametr pro spojení svislé osy.
that.isInAxesPanel (currentX,currentY)	Vrací údaj o tom, zda se vstupní souřadnice nachází v grafu (objektu AxesPanel).
that.zoom (xFrom,xTo,yFrom,yTo)	Změní rozsah svislé a vodorovné osy.
that.zoomX(xFrom,xTo)	Změní rozsah vodorovné osy.
that.zoomY(yFrom,yTo)	Změní rozsah svislé osy.
that.setAxisIcon(ax,ay)	Při zamknutí osy v objektu ChartPanel zajistí zobrazení příslušné ikony.
that.handleEvent(event)	Obstarává reakce na události myši. Dle aktivního statusu (výběr detailu, posun v grafu tažením myši) spouští při

	události *mousedown* mechanismus, který dále při *mousemove* např. vykresluje oblast vybíraného detailu (zoom), mění rozsah svislé a vodorovné osy při posunu v grafu. Událost *mousedown* rovněž ovládá zamykání a odemykání osy. Obsahuje také pravidla pro reakce na událost *mousewhell* (otáčení kolečka myši).
coordinateTransform (screen-Point,someSvgObject)	Přepočítá absolutní souřadnice vstupního parametru screenPoint na souřadnice relativní k příchozímu parametru someSvgObject.

Tabulka 6: Přehled funkcí třídy AxesPanel

Funkce třídy AxesPanel popsané v tabulce 6 budou nyní použity v příkladech.

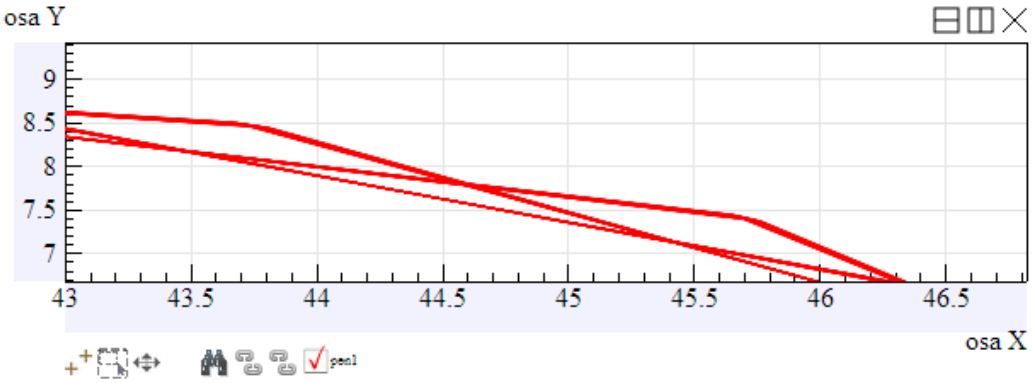


Obrázek 7: Výběr detailu v grafu

- Výběr detailu (zoom) v grafu:

  Bude rozebrána situace z obrázku 7. Uživatel klikl v menu pod grafem na ikonu pro výběr detailu (zoom), což způsobilo, že se ve třídě AxesPanel nastavil parametr mouseZoom na hodnotu *true*. Poté, když funkce that.handleEvent po kliknutí do grafu zachytila událost *mousedown* a zkontrolovala, zda má parametr mouseZoom hodnotu *true*, došlo k volání funkce coordinateTransform, kam byly poslány údaje o horizontální a vertikální souřadnici myši uložené při kliknutí a název SVG plochy, na kterou je vykreslováno. Funkce coordinateTransform převede absolutní souřadnice umístění myši ve webové stránce na souřadnice relativní k SVG ploše, na kterou je vykreslováno. Dále, vzhledem k tomu, že výběr detailu (zoom) může probíhat jak v grafu, tak pouze ve svislé či vodorovné ose, je zjišťováno, zda bylo kliknuto přímo do grafu, pod něj nebo vlevo od něj. V tomto příkladu bylo kliknuto přímo do grafu a tím se nastavil parametr drag na *true*. Když je tento parametr *true* a současně dojde k pohybu myši, funkce that.handleEvent zachytí událost *mousemove*. Dále zkontroluje, zda má parametr drag hodnotu *true* a opět volá funkci coordinateTransform pro převod souřadnic kursoru myši. Poté se nastaví parametry SVG elementu mouseSelection typu

rect, který zobrazuje uživateli oblast, kterou vybírá pro přiblížení (zoom). Na obrázku 7 je zobrazen jako světle modrý čtverec. Toto probíhá, dokud nedojde k uvolnění tlačítka myši (událost *mouseup* ve funkci that.handleEvent). Když toto nastane, přepočítají se souřadnice SVG elementu na pixely pomocí volání funkce that.value2px třídy Axis, to z důvodu, že pro následné volání funkce that.setRange třídy Axis je nutno udat požadovaný rozsah osy v jednotkách. Funkce je volána pro obě osy, aby změnila jejich rozsahy. Přesněji zavolá funkci that.setRange třídy Scale a ta změní rozsah svislé i vodorovné osy. Poslední volaná funkce se jmenuje that.updateAxesPanel, která pro všechny pera v grafu (v tomto příkladu pouze jedno pero pen1) zavolá v závislosti na tom, zda je požadováno vykreslování křivek, nebo pouze bodů, příslušné funkce that.draw, respektive that.drawPoints (tabulka 8) náležící třídě AxesPanelPen (TrendPen). Tyto funkce se postarají o vykreslení křivky (bodů) ve zvoleném rozsahu svislé a vodorovné osy. Výsledek této operace je na obrázku 8.

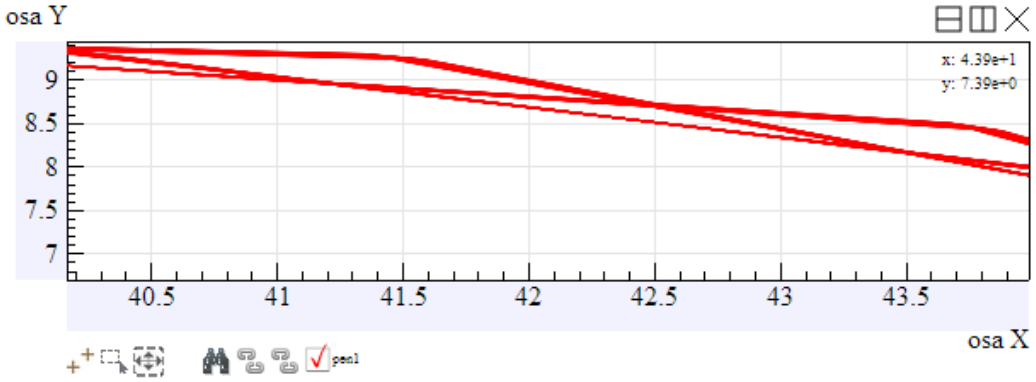


Obrázek 8: Detail v grafu

- Posun v grafu:

  V tomto příkladu je použit posun grafu, který je zpřístupněn kliknutím na příslušnou ikonu v panelu nástrojů (obrázek 3), což nastaví parametr mouseMove na *true*. Posunována je křivka z obrázku 8 o něco výše v grafu. Po kliknutí myší do grafu, dojde k podobnému procesu, jako v příkladu výše. Funkce that.handleEvent zachytí událost *mousedown*, přepočítají se absolutní souřadnice kursoru myši na relativní, pomocí funkce coordinateTransform a v případě, že má parametr mouseMove hodnotu *true*, nastaví se i parametr drag na *true* (opět lze kliknout i vlevo nebo pod graf a pohybovat se pak následně pouze v jedné ose). Když dojde k pohybu myši v grafu, nastane událost *mousemove*, kterou opět zachytí funkce that.handleEvent a v případě, že má parametr drag hodnotu *true*, dojde k volání funkce that.setRange třídy Axis pro vodorovnou i svislou osu a tím ke změně jejich rozsahů dle odpovídajících souřadnic kursoru myši. Dále je volána funkce that.updateAxesPanel, která zavolá příslušné funkce třídy

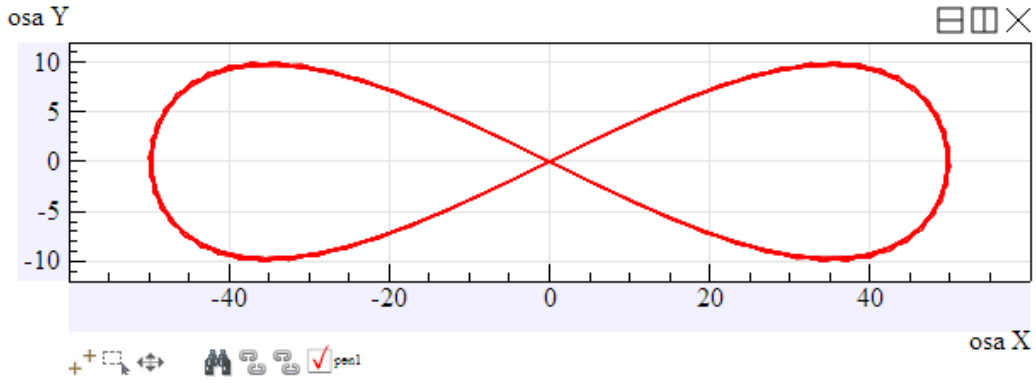
AxesPanelPen (TrendPen) pro vykreslení křivky v daném rozsahu svislé a vodorovné osy (viz předchozí příklad). Výsledek posunu je na obrázku 9.



Obrázek 9: Posun v grafu

- Příklad použití funkce that.autoScale:

  Další užitečnou a často používanou funkcí třídy je funkce that.autoScale (viz tabulku 6). Tento příklad vychází ze situace na obrázku 9, kdy po kliknutí na příslušnou ikonu v panelu nástrojů, dojde k volání funkce that.autoScale. Jejím účelem je zobrazit křivky (body) všech per v grafu v jejich maximálním rozsahu. Ve svém průběhu funkce projde tato pera a zjistí minimální a maximální hodnoty jejich horizontálních i vertikálních souřadnic. Dále jsou volány funkce that.setRange třídy Axis (viz příklad výše) ve svislé i vodorovné ose s požadavkem na změnu rozsahu každé osy. Nový rozsah je definován nejmenší a největší hodnotou nalezenou mezi pery. Z důvodu přehlednosti je respektováno odsazení od okrajů grafu (*offset*), které je v tomto příkladu nastaveno na 10 procent ve svislé i vodorovné ose. Nakonec je volána funkce that.updateAxesPanel, jež zavolá příslušné funkce třídy AxesPanelPen (TrendPen) pro vykreslení per (viz příklad výše). Výsledek této operace je na obrázku 10.



Obrázek 10: Použití funkce that.autoScale

4.6 Třídy AxesPanelPen a TrendPen

Třídy AxesPanelPen a TrendPen reprezentují pero grafu a jsou obě zděděny od třídy GraphicalObject, což jim mimo jiné přináší nutnost stejného pojmenování pro funkce vykreslující křivky a body pera. Třída AxesPanelPen je určena pro vykreslení obecných křivek umístěných v objektu AxesPanel. Třída TrendPen je určena pro vykreslení časových křivek v objektu Trend. Vytvoření instance třídy:

```
nazevPera = REX.UI.CHARTS.AxesPanelPen(settings)
```

Respektive:

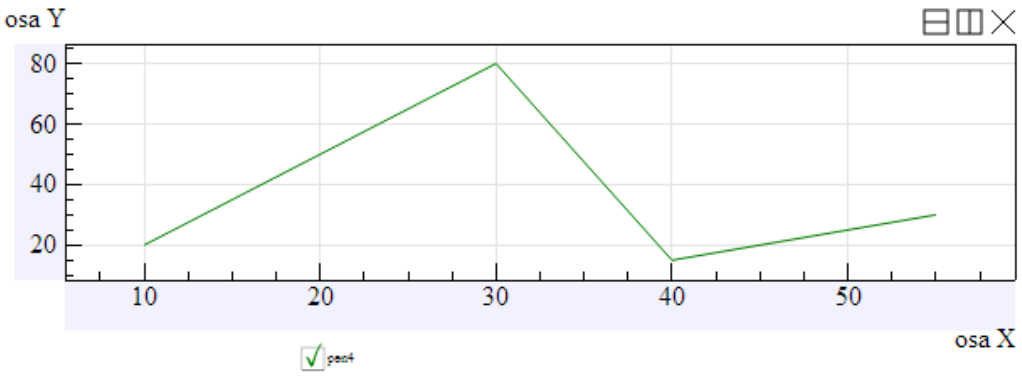
```
nazevPera = REX.UI.CHARTS.TrendPen(settings)
```

Ve vstupním objektu settings jsou předány údaje o názvu, barvě pera atd. V případě pera typu TrendPen se v objektu settings navíc definuje počáteční horizontální hodnota (tBegin), perioda (deltaT) a kapacita zásobníku (buffercapacity). Viz tabulku 7.

Parametr	Popis a možnosti nastavení
settings.name (that.name)	Název pera. Při neuvedení bude nastavena defaultní hodnota *Unnamed pen*.
settings.color (that.color)	Barva pera. Při nenastavení bude mít defaultní hodnotu *red* (červená).
settings.penwidth (that.penwidth)	Tloušťka čáry pera v pixelech. Při neuvedení bude nastavena defaultní hodnota 1.
settings.visibility (that.visibility)	Nastavení viditelnosti pera. Při hodnotě *true* bude pero vykresleno v grafu, při hodnotě *false* vykresleno nebude. Defaultně je nastavena hodnota *true*.
settings.tBegin (that.tBegin) - jen u třídy TrendPen	Horizontální hodnota prvního bodu pera třídy TrendPen. Při nenastavení hodnota 0.
settings.deltaT (that.deltaT) - jen u třídy TrendPen	Perioda vzorkování neboli vzdálenost mezi dvěma příchozími body pera TrendPen. Při neuvedení je hodnota nastavena na 1.
settings.buffercapacity (that.buffercapacity) - jen u třídy TrendPen	Kapacita zásobníku (*buffer*) pera třídy TrendPen.

Tabulka 7: Parametry objektů settings a that třídy AxesPanelPen (TrendPen)

Funkce obou tříd jsou si velmi podobné, liší se prakticky pouze v tom, že pero třídy AxesPanelPen potřebuje pro vykreslení jednoho bodu jeho vertikální i horizontální souřadnici, kdežto pero třídy TrendPen potřebuje pouze vertikální souřadnici a horizontální souřadnici si dopočte dle zadané periody. Obě funkce poskytují nástroje pro přidání bodu, či pole bodů do pera (respektive do pole, v nichž jsou uloženy souřadnice bodů), pro vymazání bodů, pro vykreslení bodů a další. Stěžejní jsou funkce that.draw a that.drawPoints zajišťující vykreslení křivek (bodů), podrobněji jsou popsány v tabulce 8 a na příkladech níže.



Obrázek 11: Pero třídy AxesPanelPen v grafu

Funkce	Popis
that.getPoints()	Vrátí pole bodů nacházejících se ve třídě AxesPanelPen.
that.getMin()	Vrátí bod s minimální vertikální i horizontální hodnotou souřadnice.
that.getMax()	Vrátí bod s maximální vertikální i horizontální hodnotou souřadnice.
that.draw(axis)	Vykreslí křivku, tak že projde všechny body objektu AxesPanelPen a spojí je přímkami (SVG element *polyline*).
that.drawPoints(axis)	Vykreslí symbol křížku (složení dvou SVG elementů typu *line* do SVG grupy) v každém umístění bodu z pole bodů, jenž náleží objektu AxesPanelPen.
that.addPoint(x,y)	Funkce pro přidání bodu do pole bodů. V třídě TrendPen se tato funkce liší tím, že do ní vstupuje pouze vertikální souřadnice a horizontální souřadnice bodu je dopočítána dle nastavené periody.
that.addArrayOfPoints (x,y)	Přidá do pole bodů třídy AxesPanelPen další pole příchozích bodů.
that.clear()	Vymaže pole bodů.

Tabulka 8: Přehled funkcí třídy AxesPanelPen

Pro lepší pochopení funkcí třídy AxesPanelPen budou obrázky 11 a 12 popsány v příkladech.

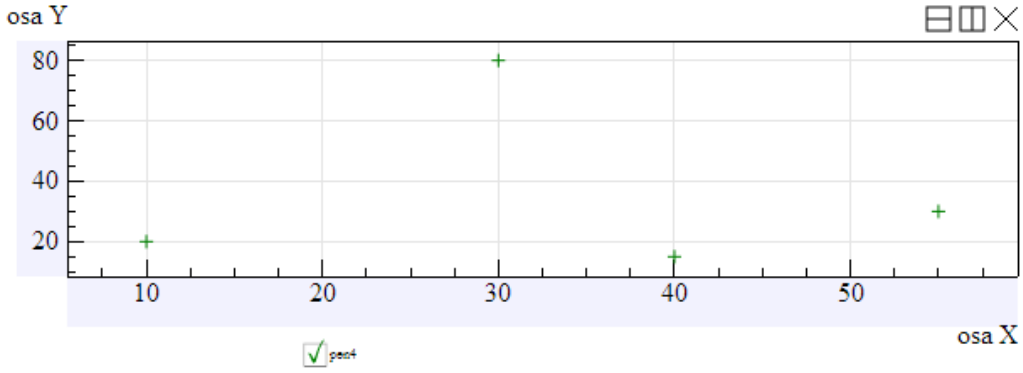
- Příklad užití funkcí třídy AxesPanelPen:

  Po vytvoření instance třídy AxesPanelPen je nutné do pera vložit body, což zajišťuje dvojice funkcí. První z nich je that.addPoint, která vloží do pera jeden bod, druhá je that.addArrayOfPoints. Ta umožňuje vložení libovolného pole bodů (viz tabulku 8). V příkladu na obrázku 11 byla pro vložení bodů do pera pen4 použita funkce that.addArrayOfPoints, jejíž konkrétní realizace v kódu vypadá takto:

```
pen4.addArrayOfPoints([10, 30, 40, 55],
                      [20, 80, 15, 30])
```

- Aby došlo k vykreslení křivky, je nutné volat funkci that.draw, do které vstupuje proměnná s referencí na vodorovnou a svislou osu. Vytvoří se SVG element typu *polyline*, do jehož parametru points jsou přidány body získané z pole bodů pera. Souřadnice bodů musí být před vložením přepočítány na hodnotu v pixelech funkcí that.value2px třídy Axis (viz tabulku 10 a příklady v 4.7). Mimo to je SVG elementu nastaveno hlídání události *mouseover*, to v tomto případě znamená, že při najetí myší na křivku dojde k zvýraznění čáry pera.

- Na obrázku 12 je zobrazeno užití funkce that.drawPoints (tabulka 10), která se od předchozího příkladu na funkci that.draw liší tím, že nevytváříme SVG element typu *polyline*, nýbrž SVG grupu složenou ze dvou elementů typu *line*. Tato grupa vytvoří „křížek“ reprezentující bod ze seznamu bodů.



Obrázek 12: Vykreslení bodů pomocí funkce that.drawPoints

4.7 Třída Axis

Třída Axis reprezentuje osu grafu. Její instance je tvořena následovně:

```
nazevosy = REX.UI.Axis(id, settings)
```

Při vytvoření je zadán název osy (id), dále v objektu settings její orientace (vertikální či horizontální), rozsah, minimum a maximum a další parametry (viz tabulku 9). Pokud není instance třídy Axis vytvářena samostatně, nýbrž je nejprve vytvořena instance třídy AxesPanel (Trend), poté se nastavení osy definuje ve vstupním objektu settings třídy AxesPanel (tabulka 5), hodnoty umístění a rozměrů osy (that.x, that.y, that.width, that.height) jsou poté nastaveny automaticky dle nastavení objektu AxesPanel (Trend).

Parametr	Popis a možnosti nastavení
settings.name (that.name)	Název osy
settings.drawaxisname (that.drawaxisname)	Zobrazení názvu osy. Při hodnotě *true* se název zobrazí, při *false* nikoliv. Defaultně je hodnota nastavena na *true*.
settings.x (that.x)	Počáteční horizontální pozice osy na webové stránce (v pixelech).
settings.y (that.y)	Viz settings.x pro vertikální pozici osy.
settings.width (that.width)	Šířka osy v pixelech.
settings.height (that.height)	Výška osy v pixelech.
settings.min (that.min)	Počáteční minimální hodnota osy.
settings.max (that.max)	Počáteční maximální hodnota osy.
settings.type (that.type)	Typ osy. Zatím dokončen typ *linear*, jenž je použit v příkladech v této práci. Další typ je *radial* (rozpracovaný).
settings.scale (that.scaletype)	Při nastavení na hodnotu *dec* vykreslujeme křivky v desítkové soustavě, při nastavení na *log* v logaritmické soustavě (tato možnost je zatím rozpracovaná).
settings.smart (that.smart)	Při nastavení na hodnotu *true* je umožněno „chytré" přepočítávání umístění čar typu *mark* a *tick*. Defaultně je parametr nastaven na hodnotu *true*.
settings.maintick (that.maintick)	Počet čar typu *mark* na ose. Při nenastavení bude mít parametr defaultní hodnotu.
settings.sectick (that.sectick)	Počet čar typu *tick* mezi dvěma čárami typu *mark*. Při nenastavení bude mít parametr defaultní hodnotu.
settings.marksize (that.marksize)	Velikost jedné čáry typu *mark* v pixelech. Při nenastavení bude mít parametr defaultní hodnotu.
settings.ticksize (that.ticksize)	Velikost jedné čáry typu *tick* v pixelech. Při nenastavení bude mít parametr defaultní hodnotu.
settings.markstyle (that.markstyle)	Při nastavení na hodnotu *in*, jsou čáry typu *mark* a *tick* vykreslovány od osy pouze směrem dovnitř grafu, při hodno-

	tě *middle* jsou vykresleny přes osu ve stejné délce vně i dovnitř grafu. Defaultně je nastavena varianta *in*.
settings.drawmarkline (that.drawmarkline)	Při hodnotě *true* vykreslí přímky v grafu, které odpovídají umístění čar typu *mark*. Defaultně je hodnota parametru nastaven jako *true*.
settings.drawtickline (that.drawtickline)	Při hodnotě *true* vykreslí přímky v grafu, které odpovídají umístění *ticků*. Defaultně je hodnota parametru nastavena jako *false*.

Tabulka 9: Nastavitelné parametry objektů settings a that třídy Axis

Osa (třída Axis) si vytvoří instanci třídy Scale, která slouží jako nástroj udávající funkcím třídy Axis veškeré údaje, které potřebují pro vykreslení osy. To znamená, že třída Scale udává zobrazovaný rozsah (minimum a maximum), počet čar typů *mark* a *tick*, hustotu popisků atd.

Nejdůležitější funkcí třídy Axis je drawAxis, která zprostředkovává vykreslení veškerých grafických prvků osy. Způsob jejího fungování je vysvětlen v příkladu níže. Kromě vykreslení osy poskytuje třída Axis nástroje vyšší třídě AxesPanel (Trend), pomocí níž můžeme měnit např. rozsah osy, nebo přepočítávat údaj v pixelech na hodnotu odpovídající rozsahu osy (viz tabulku 10).

Zamykání osy, jež je ovládané z vyšší třídy AxesPanel nebo Trend má následující princip. V případě zamknutí osy, která náleží třídě AxesPanel zcela zamezíme jakémukoliv měnění jejího rozsahu. V případě třídy Trend pro časové křivky, zamknutím osy vypneme možnost volání funkce that.autoScale (tabulka 6), to nám ve výsledku umožní pozorovat pouze část rozsahu osy „běžícího“ grafického objektu Trend (viz obrázek 15 v 4.9.2).

Funkce	Popis
that.px2value(px)	Přepočítá vstupní hodnotu v pixelech na hodnotu odpovídající reálné hodnotě v grafu.
that.value2px(val)	Přepočítá vstupní proměnou value z grafu na údaj v pixelech.
that.setRange(min, max)	Zavolá příslušné funkce třídy Scale a změní rozsah osy.
that.setJustRange (min, max)	Zavolá příslušné funkce třídy Scale a změní rozsah osy. Od funkce that.setRange se liší jen tím, že se nezabývá problémem spojení vodorovné či svislé osy.
that.setMax(max)	Zavolá příslušné funkce třídy Scale a změní maximum osy.
that.setMin(min)	Zavolá příslušné funkce třídy Scale a změní minimum osy.
that.scaleaxis(s)	Zavolá příslušné funkce třídy Scale a změní rozsah osy.
drawAxisName()	Přidá k ose její název.
drawMark(x, y)	Vykreslí jedu čáru typu *mark* neboli větší čáru na ose.
drawTick(x, y)	Vykreslí jednu čáru typu *tick*, neboli menší čáru na ose. *Tick* se nachází mezi dvěma čárami typu *mark*.
drawAxis()	Nejdůležitější funkce této třídy, jež se stará o postupné volání příslušných funkcí potřebných k vykreslení všech grafických prvků osy.

drawsmartlabel (value, position)	Vytvoří popisek osy na základě jeho hodnoty a polohy.
createTextElement(text)	Funkce vrací text ve formě SVG elementu.

Tabulka 10: Přehled funkcí třídy Axis

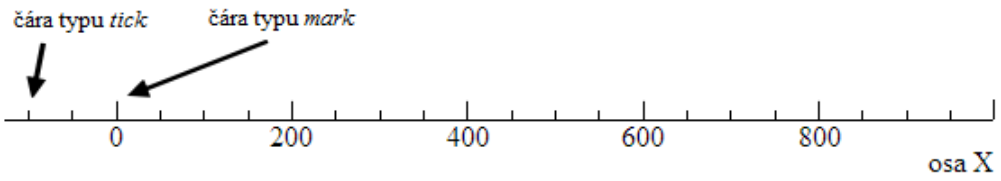
K pochopení funkce třídy Axis pomůže podrobnější pohled na funkci drawAxis.

- Princip funkce drawAxis:

  Při volání funkce drawAxis se nejprve vytvoří SVG grupy pro uložení grafických prvků osy. Postupně se vytvoří hlavní čára osy, na kterou se nejprve umístí čáry typu *mark*, které jsou vytvořeny cyklickým voláním funkce drawMark, při čemž počet volání této funkce se řídí parametrem that.maintick určující počet čar typu *mark* na ose.

  Tento počet je nastaven při vytváření instance třídy Axis ve vstupním objektu settings. Umístění čar typu *mark* je přepočítáváno parametry that.a0mark a that.damark třídy Scale (viz. 4.8). Tyto údaje přicházejí v číselných hodnotách odpovídajících rozsahu osy. Pro vykreslení čáry typu *mark* je použit SVG element typu *line*. Hodnota, udávající jeho polohu musí být přepočítána na údaj v pixelech pomocí funkce that.value2px. Při každém volání funkce drawMark dochází současně k volání funkce drawsmartlabel pro vytvoření číselných popisků nad čárami typu *mark*. Dalším krokem je vytvoření čar typu *tick* cyklickým voláním funkce drawTick, přičemž počet čar typu *tick* je určen parametrem that.sectick, jeho hodnota je nastavena vstupním objektem settings. Umístění čar typu *tick* je určeno třídou Scale, respektive jejími parametry that.n0tick a that.datick.

  Na obrázku 13 je konkrétní implementace třídy Axis, kde je parametr that.maintick roven šesti a parametr that.sectick je pět (dochází k překrytí). Rovněž si můžeme povšimnout popisků pod čárami typu *mark* vytvořenými funkcí drawsmartlabel.



Obrázek 13: Implementace třídy Axis

4.8 Třída Scale

Instance třídy Scale je automaticky vytvořena pro každou třídu Axis při jejím vzniku a určuje této třídě, v jakém rozsahu bude vykreslována, kolik čar typu *mark* a *tick* na ní bude a kde se budou fyzicky nacházet. Instance třídy Scale je vytvořena ve tvaru:

```
nazevScale = REX.UI.Axis.Scale(settings)
```

Ve vstupním objektu settings jsou předány především informace o počtu čar typu *tick* a *mark*, minimu a maximu osy atd. Třída obsahuje dvě důležité funkce (viz tabulku 11). That.setRange, jež mění rozsah osy (popř. počet čar typu *tick* a *mark*) a setmarkingpars, která přepočítává umístnění čar typu *mark* a *tick*. Pro lepší pochopení této funkce je níže uveden příklad jedné její implementace.

Funkce	Popis
that.setRange (min,max,nmarks,nticks,smart)	Změní rozsah osy dle vstupních proměnných min a max. Dále změní počet čar typu *mark* na ose dle parametru nmarks a počet čar typu *tick* dle parametru nticks. Dále dle vstupní proměnné smart lze zapnout, či vypnout „chytré“ přepočítávání umístění čar typů *mark* a *tick*, které zajišťuje funkce setmarkingpars.
setmarkingpars()	„Chytré“ přepočítávání umístění i počtu čar typů *mark* a *tick* tak, aby čára typu *mark* byla vždy u nějaké zaokrouhlené hodnoty (desítky, stovky atd.). Výpočet vzdálenosti mezi dvěma čárami obou typů.

Tabulka 11: Přehled funkcí třídy Scale

- Implementace funkce setmarkingpars:

  Na obrázku 13 je zobrazen výsledek činnosti funkce setmarkingpars, která nejprve spočítá dle minima a maxima osy její rozsah a dále dle určeného počtu čar typů *mark* a *tick* provede výpočet, jehož výsledky si popíšeme. Parametr that.a0tick udávající umístnění první čáry typu *tick* je –20. Parametr that.a0mark udávající umístění první čáry typu *mark* je 0. Vzdálenost mezi dvěma čarami typu *tick* definovaná parametrem that.datick je 50, vzdálenost mezi dvěma čarami typu *mark* definovaná parametrem that.damark je 200. Tyto údaje jsou nezbytné pro funkci drawAxis z třídy Axis, která musí ještě před tím než začne vykreslovat provést přepočet hodnot získaných z funkce setmarkingpars na pixely pomocí funkce that.value2px.

4.9 Příklad použití grafického objektu ChartPanel pro vykreslování časových křivek

V předchozích kapitolách byly dle hierarchie podrobněji popsány třídy grafického objektu ChartPanel. Následuje ukázka na příkladu, demonstrující jak v prostředí webového prohlížeče vytvořit instanci grafického objektu ChartPanel, který bude obsahovat jednu instanci třídy Trend pro vykreslování časových křivek, jejichž body budou postupně generovány a vkládány do dvou per třídy TrendPen.

4.9.1 Postup vytvoření instancí jednotlivých tříd grafického objektu

Prvním krokem je vložení následujících prvků do hlavičky webové stránky:

```
<link rel="stylesheet" type="text/css" href="css/rex-trend-style.css"/>

<script src="js/jquery-1.7.1.min.js"></script>
<script type="text/javascript" src="js/rex-base.js"> </script>
<script type="text/javascript" src="js/rex-ui-charts.js"> </script>
```

Tímto byly načteny kaskádové styly (css) náležící grafickému objektu, dále byl načten soubor se skripty z knihovny jQuery [14], jejichž několik funkcí budeme níže používat. Dále byly načteny *js* (JavaScript) soubory rex-base, s pomocnými funkcemi, a rex-ui-charts s třídami a funkcemi grafického objektu ChartPanel.

V těle webové stránky bude vytvořen SVG prvek, kterému bude přiřazeno id trendholder a nastavena šířka 860 pixelů a výška 760 pixelů. Tento SVG element bude později sloužit jako „plátno“ do kterého bude umístněn grafický objekt.

```
<svg id="trendholder" style="width:860px; height:760px;"></svg>
```

Vlastní skript inicializace bude vložen do následujícího rámce:

```
<script type="text/javascript">
      $(document).ready(function () {
            //Zde budu inicializovat grafický objekt ChartPanel
      });
</script>
```

Zde je využita funkce ready z knihovny jQuery, která zajistí vykonání skriptu až po načtení všech prvků webové stránky.

Nyní již bude přikročeno k samotnému vytvoření instance třídy ChartPanel:

```
mujChartPanel = REX.UI.CHARTS.ChartPanel('casovy_ChartPanel',
                {
                    drawToolbar: true
                });
```

Instance třídy ChartPanel je pojmenována mujChartPanel. Jako id nastaven název casovy_ChartPanel a v objektu settings nastavena pouze vlastnost drawToolbar na hodnotu *true*, což znamená, že je povoleno vykreslení panelu nástrojů.

Nyní bude vytvořena instance třídy Trend, která bude pojmenována muj_trend. Parametru id bude nastaven název casovy_trend a dále budou zadány vlastnosti objektu settings, tedy výška (width), šířka (height), odsazení (margin) a umístnění (x, y) grafu. V objektu settings rovněž budou nastaveny parametry objektů x_axis a y_axis, ze kterých se při vytváření instance třídy Trend vytvoří osy neboli instance třídy Axis.

U vodorovné i podélné osy bude nastaveno její jméno (name) a jeho zobrazení (drawaxisname). U vodorovné osy bude orientace (orientation) nastavena jako horizontální (horizontal), u svislé osy bude nastavena jako vertikální (vertical). Typ osy bude lineární (linear) a soustava (scale) bude desítková (dec). Dále budou nastavena minima (min) a maxima (max) osy a parametr smart na hodnotu *true*, což znamená, že se umístění čar typů *tick* a *mark* bude „chytře“ přepočítávat (viz tabulku 11 a příklad u třídy Scale vysvětlující princip funkce setmarkingpars).

```
muj_trend = REX.UI.CHARTS.Trend('casovy_trend',
                {
                    width: 700,
                    height: 200,
                    x: 80,
                    y: 440,
                    margin: { l: 10, t: 10, b: 10, r: 20 },
 x_axis: {
            name: 'cas [s]',
            drawaxisname: true,
            orientation: 'horizontal',
            type: 'linear',
            drawtickline: false,
            drawlabel: true,
            scale: 'dec',
            min: -150,
            max: 1000,
            smart: true
        },
y_axis: {
            name: 'a*sin(10*t)',
            orientation: 'vertical',
            type: 'linear',
            drawtickline: false,
            drawlabel: true,
            scale: 'dec',
            min: -40,
            max: 260,
            smart: true
      }
});
```

Nyní bude použita funkce append z knihovny jQuery pro vložení vytvořené instance třídy ChartPanel pojmenované mujChartPanel do SVG „plátna“ trendholder.

```
$('#trendholder').append(mujChartPanel.component);
```

Do instance třídy ChartPanel bude vložena výše vytvořená instance třídy Trend pomocí funkce addAxesPanel (viz tabulku 4).

```
mujChartPanel.addAxesPanel(trend);
```

Dalším krokem je vytvoření dvou per, tedy instancí třídy TrendPen. Budou nastavena jejich jména (name) na pen_1 a pen_2. Dále budou nastaveny barvy (color) per a jelikož se jedná o instanci třídy TrendPen pro vykreslování časové křivky, je nutno ještě nastavit počáteční hodnotu vodorovné osy (tBegin), periodu vzorkování (deltaT) a kapacitu bufferu (buffercapacity).

```
pen_1 = REX.UI.CHARTS.TrendPen
({
      name: 'pen1',
      color: 'orange',
      tBegin:0,
      deltaT:10,
      buffercapacity:400
});

pen_2 = REX.UI.CHARTS.TrendPen
({
      name: 'pen2',
      color: 'red',
      tBegin:0,
      deltaT:10,
      buffercapacity:400
});
```

Následně budou přidána do grafu pera pomocí funkce that.addPen (viz tabulku 6).

```
muj_trend.addPen(pen_1);
muj_trend.addPen(pen_2);
```

Nyní budou použity funkce z třídy ChartPanel (viz tabulku 4) pro vykreslení panelu nástrojů (that.drawToolBar), horní lišty nad grafickým objektem (that.drawUpToolBar) a ikon per (that.drawPensMarks).

```
mujChartPanel.drawToolBar();
mujChartPanel.drawUpToolBar();
mujChartPanel.drawPensMarks();
```

Funkce that.updateChartPanel (viz tabulku 4) zavolá ve všech objektech AxesPanel (Trend) umístěných v objektu ChartPanel příslušné funkce pro vykreslení křivek jednotlivých per. V tomto případě pouze v jednom objektu Trend.

```
dmujChartPanel.updateChartPanel();
```

Hodnoty, které přicházejí ze serveru webových soketů (nebo generátoru), jsou do per vkládány pomocí funkce that.addPoint (viz tabulku 8).

```
pen_1.addPoint(hodnota);
pen_2.addPoint(hodnota_2);
```

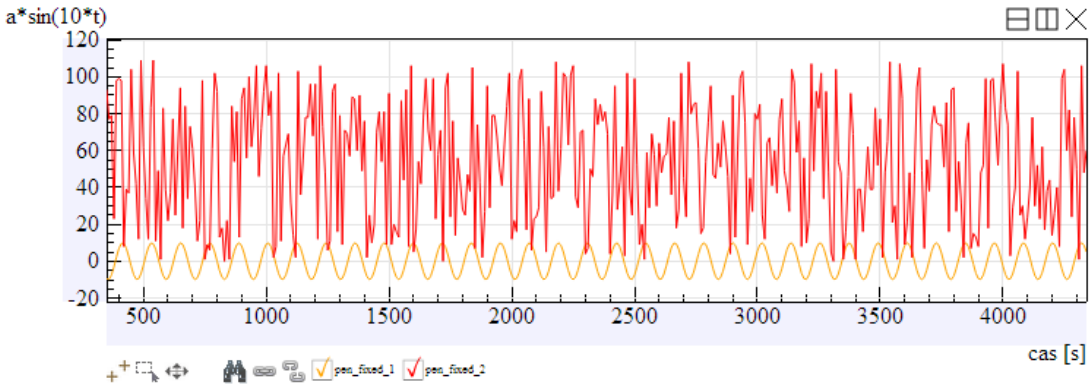
V tomto případě přichází pouze hodnota polohy ve svislé ose, jelikož pera jsou instancemi třídy TrendPen a hodnota polohy ve vodorovné ose je dopočítávána dle nastavení periody. Pokud by byla používána obecná křivka, instance třídy AxesPanelPen, musela by do funkce addPoint přicházet hodnota polohy příchozího boduve vodorovné i svislé ose.

Po každém přidání bodu se opět volá funkce that.updateChartPanel (viz tabulku 4), která ve všech svých objektech typu AxesPanel (Trend) volá příslušné funkce pro vykreslení křivky pera.

```
mujChartPanel.updateChartPanel();
```

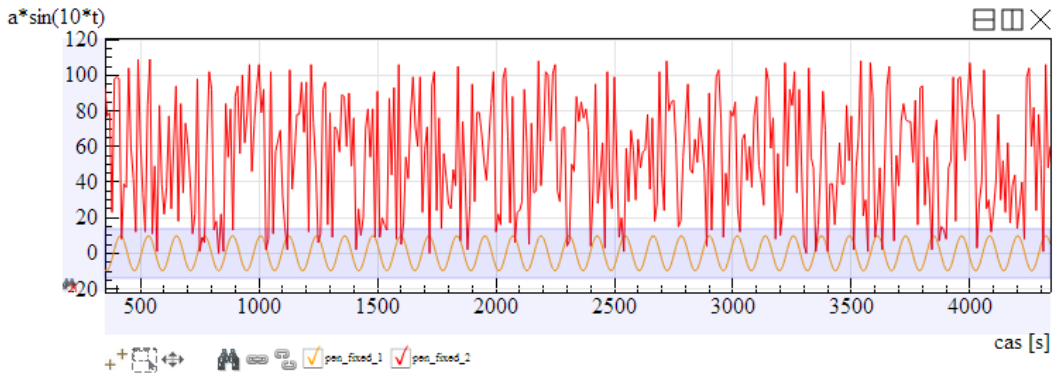
4.9.2 Příklady použití vytvořeného grafického objektu

Na obrázku 14 je grafický objekt ChartPanel „v běhu“. Objekt s jedním grafem, jež je typu Trend, byl nadefinován v 4.9.1.



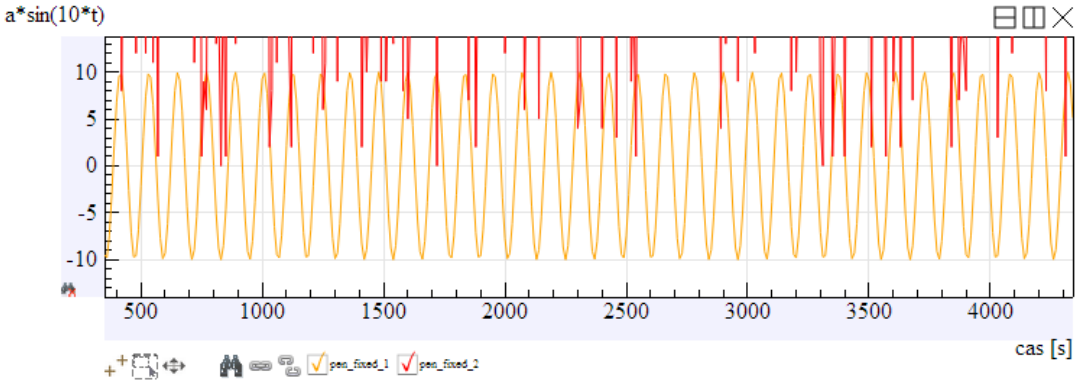
Obrázek 14: Aplikace grafického objekt ChartPanel s jednou instancí třídy Trend

Při každém přidání bodu je volána funkce that.updateChartPanel (viz 4.9.1), která v případě časového grafu volá funkci that.autoScale z třídy Trend. Ta vždy zajistí zobrazení křivek v jejich maximálním rozsahu. Z toho vyplývá, pokud uživatel chce v grafu provést výběr detailu (zoom) a přiblížit si například jednu z křivek, jak je naznačeno v příkladu na obrázku 15, musí nejprve zamknout osu.



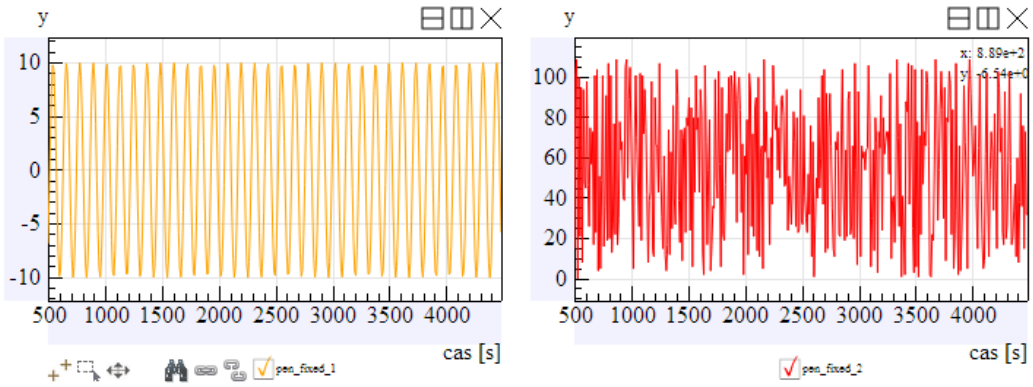
Obrázek 15: Ukázka zamknutí osy grafického objektu Trend

Na obrázku 16 je vidět, že poté co byl proveden výběr detailu (zoom) a osa je zamknutá, Trend stále zobrazuje uživatelem zvolenou oblast, přestože přichází nové body (nedochází k vykonávání funkce that.autoScale z třídy AxesPanel).



Obrázek 16: Trend - zamknutá osa a výběr detailu

Další úpravou, kterou by uživatel mohl provést je horizontální rozdělení (*split*) grafu (obrázek 17) a následné přetažení myší jednoho signálu do nově vzniklého grafu. Tím je dosaženo toho efektu, že uživatel vidí přehledně oba signály, které se zobrazují v jejich ideálních rozsazích. Vodorovná osa je u třídy Trend defaultně nastavena jako spojená (ikona řetězu), čili se ve všech instancích třídy Trend v objektu ChartPanel mění ve stejném rozsahu.



Obrázek 17: Rozdělení (split) grafického objektu trend

5 Ostatní grafické objekty

Tato kapitola je věnována tvorbě grafických objektů, které ve virtuálních laboratořích doplní funkci grafického objektu ChartPanel. Grafický objekt Bargraf se stane součástí knihovny vizualizačních nástrojů REX.UI.CHARTS a grafické objekty Slider a Display budou zařazeny do knihovny REX.UI.

5.1 Grafický objekt BarGraf

Bargraf je vizualizační prvek sloužící ke grafickému znázornění vstupní hodnoty pomocí barevného sloupce (sloupcový graf). V rámci této práce byl vytvořen horizontální i vertikální bargraf. Třída je zařazena do skupiny nástrojů REX.UI.CHARTS a její instance se vytváří následujícím zápisem:

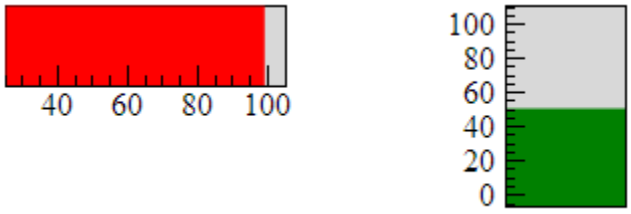
```
REX.UI.CHARTS.BarGraf = function(id, settings)
```

Ve vstupním parametru id je zadán název instance třídy BarGraf a ve vstupním objektu settings přichází údaje o rozměrech, souřadnicích umístění na webové stránce, rozsazích zobrazované osy, kritických hodnotách atd. Více o možnostech nastavení lze nalézt v tabulce 12. Při vytváření instance třídy Bargraf je dle nastavené orientace vytvořena horizontální či vertikální osa neboli instance třídy Axis. Tento grafický objekt obsahuje jedinou veřejnou funkci that.setValue (viz tabulku 13) zpřístupňující vstup hodnoty udávající výšku hladiny zobrazovaného sloupce.

Parametr	Popis a možnosti nastavení
settings.orientation (that.orientation)	Orientace grafického prvku Bargraf. Dle ní bude obsahovat vertikální nebo horizontální osu (instanci třídy Axis)
settings.x(that.x)	Horizontální pozice horního levého rohu objektu Bargraf.
settings.y(that.y)	Vertikální pozice horního levého rohu objektu Bargraf.
settings.width(that.width)	Šířka objektu Bargraf
settings.height(that.height)	Výška objektu Bargraf
settings.min(that.min)	Počáteční minimální hodnota osy.
settings.max(that.max)	Počáteční maximální hodnota osy.
settings.saveMin (that.saveMin)	Spodní „bezpečná“ hodnota osy. Po vstupu hodnoty nižší, než je tato, dojde ke změně barvy sloupce na barvu udanou dle parametru that.warningColor.
settings.saveMax (that.saveMax)	Horní „bezpečná“ hodnota osy. Po vstupu hodnoty vyšší, než je tato, dojde ke změně barvy sloupce na barvu udanou dle parametru that.warningColor.
settings.color(that.color)	Barva sloupce, jehož výška je udána vstupní hodnotou.
settings.backgroundColor (that.backgroundColor)	Barva pozadí.
settings.warningColor (that.warningColor)	Barva sloupce při vstupu hodnot nižších, než mez udaná vstupem settings.saveMin a hodnot vyšších, než mez udaná hodnotou settings.saveMax.

settings.colorIntensity (that.colorIntensity)	Intenzita barvy sloupce.
settings.backgroundColorInt (that.backgroundColorInt)	Intenzita barvy pozadí.
settings.margin (that.margin)	Objekt definující odsazení od okrajů objektu Bargraf, jež jsou definovány parametry that.x a that.y. Nastavují se hodnoty: • Odsazení zleva – l, • odsazení zprava – r, • odsazení shora - t, • odsazení zdola – b.

Tabulka 12: Nastavitelné parametry objektů settings a that třídy BarGraf



Obrázek 18: Horizontální a vertikální verze objektu BarGraf

Funkce	Popis
create()	Vytvoří SVG grupy pro umístění grafických prvků objektu BarGraf.
that.setValue(value)	Dle vstupní hodnoty value nastaví a vykreslí sloupec v příslušné výšce a barvě.

Tabulka 13: Funkce třídy Bargraf

Na obrázku 18 je ukázka horizontální a vertikální verze objektu Bargraf. Pro lepší pochopení funkcí a nastavení objektu Bargraf bude situace z obrázku rozebrána na příkladu níže.

- Zobrazení hodnoty funkcí that.setValue:

  Vlevo na obrázku 18 se nachází instance třídy BarGraf, jejíž hodnota parametru that.orientation je nastavena jako *horizontal*, parametry pro rozsah osy jsou poté nastaveny následovně. Parametr that.min má hodnotu 25 a parametr that.max 105. Parametry udávající spodní a horní „bezpečné“ hodnoty v ose jsou poté nastaveny následujícím způsobem. Parametr that.saveMin má hodnotu 5 a that.saveMax je nastaven na 99. Objekt Bargraf vpravo má hodnotu that.orientation nastavenou na *vertical*. Parametry udávající rozsah osy a její „bezpečné“ hodnoty jsou nastaveny následovně. Parametr that.min má hodnotu

-5, that.max je 110, that.saveMin je zadán jako 20 a that.saveMax je 100. Pro oba objekty Bargraf byl shodně nastaven parametr that.color, udávající barvu sloupce na hodnotu *green* (zelená) a parametr that.warningColor, popisující barvu v mezních situacích na *red* (červená). Do objektu BarGraf vlevo vstupuje pomocí funkce that.setValue hodnota 99,17. Jedná se o hodnotu vyšší, než je hodnota uvedená parametrem that.saveMax, proto je barva sloupce červená (*red*). Do instance třídy BarGraf vpravo vstupuje hodnota 49,5, která nepřekračuje žádné limity a barva sloupce je proto zelená (*green*).

5.2 Grafický objekt Slider

Posuvník neboli slider je grafický prvek, který umožňuje zadání hodnoty posunem jezdce v ose pomocí myši. Třída Slider je součástí skupiny nástrojů REX.UI. Pro účely virtuálních laboratoří byla vytvořena horizontální i vertikální verze třídy Slider. Její instance se vytváří následujícím zápisem:

```
REX.UI.Slider = function(id, settings)
```

Ve vstupním parametru id je zadán název instance třídy Slider a ve vstupním objektu settings přichází údaje o rozměrech, souřadnicích umístění na webové stránce, rozsazích zobrazované osy a další. Více informací o vstupních parametrech lze nalézt v tabulce 14. Pokud nebudou některé parametry nastaveny uživatelem, dojde při vytváření instance třídy Slider k jejich automatickému vygenerování. Pro základní nastavení postačí uživateli nastavit orientaci, umístnění a rozměry objektu Slider. Třída Slider má pouze obsahuje funkce popsáné v tabulce 15.

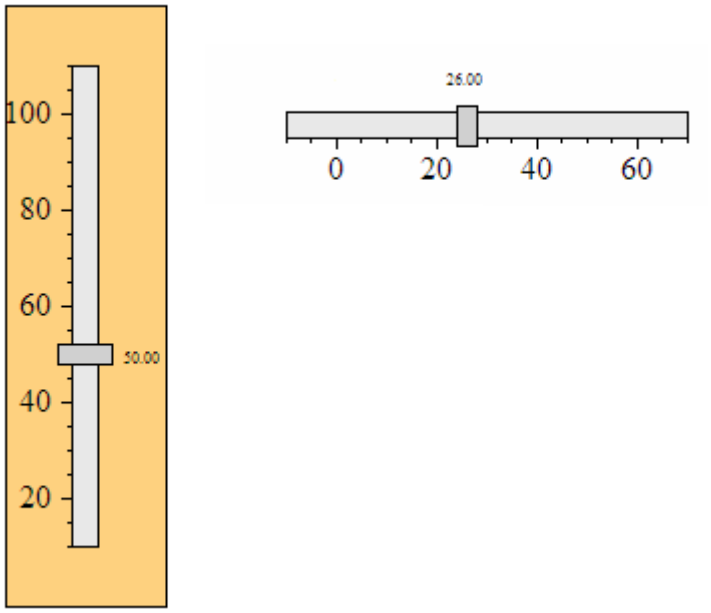
Parametr	Popis a možnosti nastavení
settings.orientation (that.orientation)	Orientace grafického prvku Slider. Dle ní bude Slider obsahovat vertikální respektive horizontální osu (instanci třídy Axis)
settings.x(that.x)	Horizontální pozice horního levého rohu objektu Slider.
settings.y(that.y)	Vertikální pozice horního levého rohu objektu Slider.
settings.width(that.width)	Šířka objektu Slider.
settings.height(that.height)	Výška objetu Slider.
settings.min(that.min)	Počáteční minimální hodnota osy.
settings.max(that.max)	Počáteční maximální hodnota osy.
settings.backgroundColor (that.backgroundColor)	Barva pozadí.
settings.trackColor (that.trackColor)	Barva lišty, po níž je pohybováno jezdcem.
settings.thumbColor (that.thumbColor)	Barva jezdce, jež udává námi zvolenou hodnotu.
settings.thumbActiveColor (that.thumbActiveColor)	Barva při najetí myší na jezdec.
settings.backgroundColorInt (that.backgroundColorInt)	Intenzita barvy pozadí.

settings.value (that.value)	Hodnota zvolená posouváním myší. Ve vstupním settings.value lze nastavit počáteční hodnotu.
settings.margin (that.margin)	Objekt definující odsazení od okrajů objektu Slider, které jsou definovány parametry that.x a that.y. Nastavují se hodnoty: • Odsazení zleva – l, • odsazení zprava – r, • odsazení shora - t, • odsazení zdola – b.
settings.trackOffset	Odsazení lišty od okraje objektu.
settings.trackWidth	Šířka lišty.
settings.trackHeight	Výška lišty.
settings.thumbWidth	Výška jezdce.
settings.thumbHeight	Šířka jezdce.
settings.backgroundBorderSz	Velikost ohraničení objektu Slider.

Tabulka 14: Nastavitelné parametry objektů settings a that třídy Slider

Funkce	Popis
create()	Vytvoří SVG grupy pro umístění grafických prvků objektu Slider.
that.moveThumb(value)	Zajišťuje posun jezdce objektu Slider.
that.handleEvent(event)	Pokud dojde k stisknutí myši nad jezdcem objektu Slider, hlídá výskyt události *mousemove*, během kterého volá funkci that.moveThumb. Rovněž hlídá výskyt události *mouseup* pro zafixování jezdce v dané poloze.

Tabulka 15: Funkce třídy Slider



Obrázek 19: Vertikální a horizontální verze objektu Slider

Pro lepší pochopení možností nastavení objektu Slider (viz tabulku 14) bude podrobněji popsána situace z obrázku 19.

- Nastavení objektu Slider:

  Vlevo na obrázku 19 je instance třídy Slider, kde je parametr that.orientation, udávající orientaci objektu Slider a jeho osy (instance třídy Axis), nastaven na hodnotu *vertical*. Výška objektu reprezentovaná parametrem that.height je 300. Odsazení lišty od okraje zadané pomocí parametru settings.trackOffset má hodnotu 10 a výška lišty popsaná parametrem settings.trackHeight) je 280. Parametr that.backgroundColor, udávající barvu pozadí, je nastaven na hodnotu *orange* (oranžová) a intensita barvy, která je určená parametrem that.backgroundColorInt, má hodnotu 0,5. Rozsah osy je určený parametry that.min, který je zde na hodnotě 10, a that.max, jenž má hodnotu 110. Hodnota parametru that.value je 50. Vpravo na témže obrázku je instance třídy Slider, která má parametr that.orientation nastaven na hodnotu *horizontal*. Parametr that.backgroundColor má hodnotu *white* (bílá) a šířka okraje objektu (settings.backgroundBorderSz) je 0. Z toho vyplývá, že na bílém pozadí nebudou v tomto případě hranice objektu zřetelné. Hodnota parametru that.value je 26.

5.3 Grafický objekt Display

Grafický objekt Display slouží jako nástroj pro zobrazení příchozí číselné hodnoty. Třída Display je součástí skupiny nástrojů REX.UI. Její instance se vytváří následujícím zápisem:

```
REX.UI.Display = function(id, settings)
```

Ve vstupním parametru id vstupuje název instance třídy Display a v objektu settings je definováno počáteční nastavení. Nastavuje se například velikost písma, či počet desetinných míst zobrazované hodnoty. Více o možnostech nastavení objektu Display lze nalézt v tabulce 16. Jeho funkce jsou poté popsány v tabulce 17.

Parametr	Popis a možnosti nastavení
settings.x(that.x)	Horizontální pozice horního levého rohu objektu Display.
settings.y(that.y)	Vertikální pozice horního levého rohu objektu Display.
settings.width(that.width)	Šířka objektu Display.
settings.height(that.height)	Výška objetu Display.
settings.backgroundColor (that.backgroundColor)	Barva pozadí.
settings.displayColor (that. displayColor)	Barva displeje (plochy, kde se zobrazuje číselná hodnota).
settings.backgroundColorInt (that.backgroundColorInt)	Intenzita barvy pozadí.
settings.decimalCount (that.decimalCount)	Počet desetinných míst zobrazované číselné hodnoty.
settings.fontSize (that.fontSize)	Velikost písma zobrazované číselné hodnoty.
settings.value	Počáteční zobrazená číselná hodnota.
settings.margin(that.margin)	Objekt definující odsazení od okrajů objektu Display, které jsou definovány parametry that.x a that.y. Nastavovanými hodnotami jsou: • Odsazení zleva – l, • odsazení zprava – r, • odsazení shora - t, • odsazení zdola – b.

Tabulka 16: Nastavitelné parametry objektů settings a that třídy Display

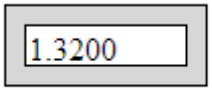
Funkce	Popis
create()	Vytvoří SVG grupy pro umístění grafických prvků objektu Display.
that.setValue(value)	Dle vstupní hodnoty value zobrazí číselnou hodnotu.

Tabulka 17: Funkce třídy Display

Použití funkce that.setValue s ohledem na nastavení bude vysvětleno na jednoduchém příkladu.

- Zobrazení číselné hodnoty:

  Na obrázku 20 se nachází instance třídy Display, do níž pomocí funkce that.setValue vstoupila hodnota (value) 1,32. Číselná hodnota je dle nastavení parametru that.decimalCount zobrazena na čtyři desetinná čísla.



Obrázek 20: Grafický objekt Display

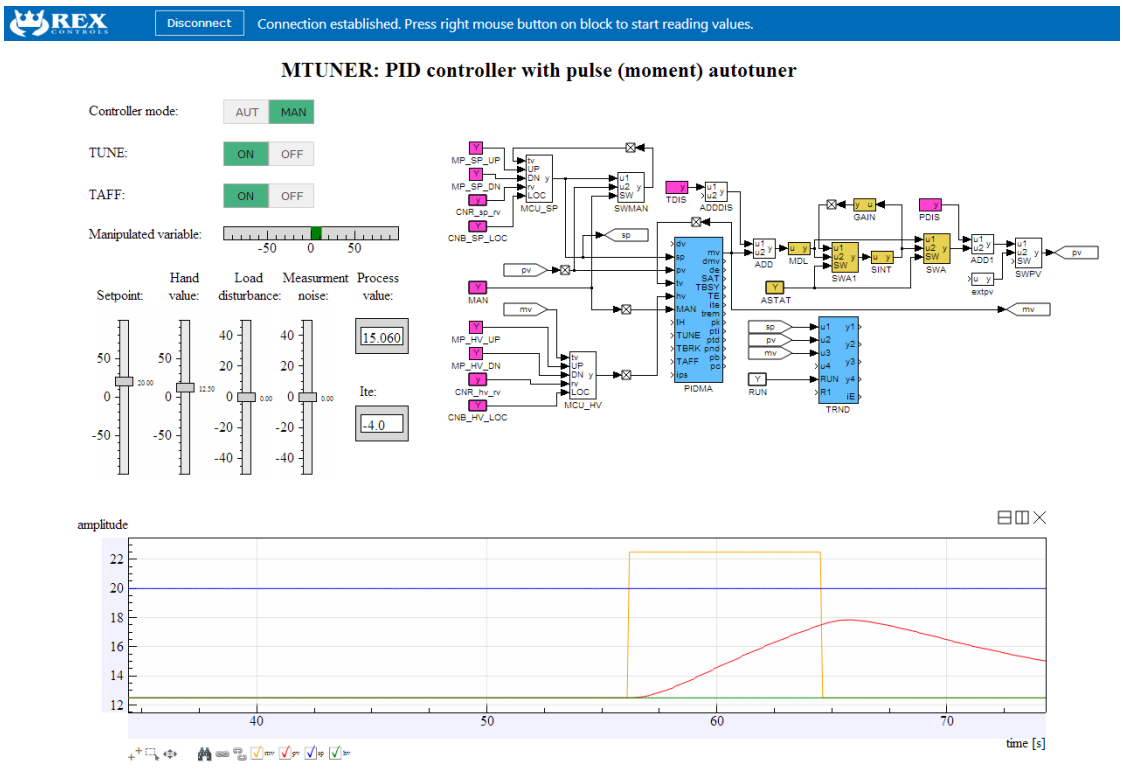
6 Vytvořené virtuální laboratoře

Předchozí kapitoly byly věnovány postupnému vývoji komponent, které nyní mohou být využity při vytváření úloh virtuálních laboratoří. V těchto úlohách bude prezentována funkčnost těchto komponent. Prezentované příklady byly vytvořeny v řídicím systému REX, jsou součástí jeho instalace a slouží jako demonstrace možností tohoto systému.

Při tvorbě virtuálních laboratoří je využito možnosti programu RexDraw uložit schémata funkčních bloků ve formátu SVG. To zpřístupňuje možnost vložení schémat do webové stránky. Poté je pomocí jazyka JavaScript nastavena komunikace se serverem webových soketů řídicího systému REX, který umožní výměnu dat s jádrem RexCore. Dalším krokem je nastavení propojení toku dat s vloženými grafickými objekty. Vytvořené virtuální laboratoře využívají všechny komponenty vytvořené v rámci této práce – grafické objekty ChartPanel, BarGraf, Display a Slider.

6.1 MTUNER: PID regulátor s momentovým autotunerem

Virtuální laboratoř pro ladění PID regulátoru s momentovým autotunerem (*PID controller with pulse moment autotuner*) z obrázku 21 je založena na schématu převzatém z demonstračních příkladů, které jsou součástí instalace řídicího systému REX.



Obrázek 21: Virtuální laboratoř pro ladění PID regulátoru s momentovým autotunerem

6.1.1 Popis příkladu

Základní komponentou tohoto příkladu je výše zmíněný PID regulátor s momentovým autotunerem neboli PIDMA [4] z podknihovny REG knihovny RexLib řídicího systému REX (viz tabulku 2). PIDMA rozšiřuje funkce klasického PID regulátoru o možnost automatického nastavení parametrů. Pro správné provedení identifikačního experimentu, jenž se spouští vstupem TUNE, musí operátor ve vhodném pracovním bodě dosáhnout ustáleného stavu a poté zvolit typ regulátoru. Aby provedený experiment mohl změnit parametry regulátoru, musí být vstup TAFF nastaven na hodnotu *on*. Algoritmus ladění nejprve odhadne šum a drift měření, následně je vygenerován obdélníkový puls na vstup procesu. Z odezvy procesu jsou vypočítány první tři momenty impulsní funkce. Na základě těchto momentů je odhadnut model procesu, jenž je použit pro automatické nastavení parametrů regulátoru. Na obrázku 21 je možné pozorovat probíhající identifikační experiment.

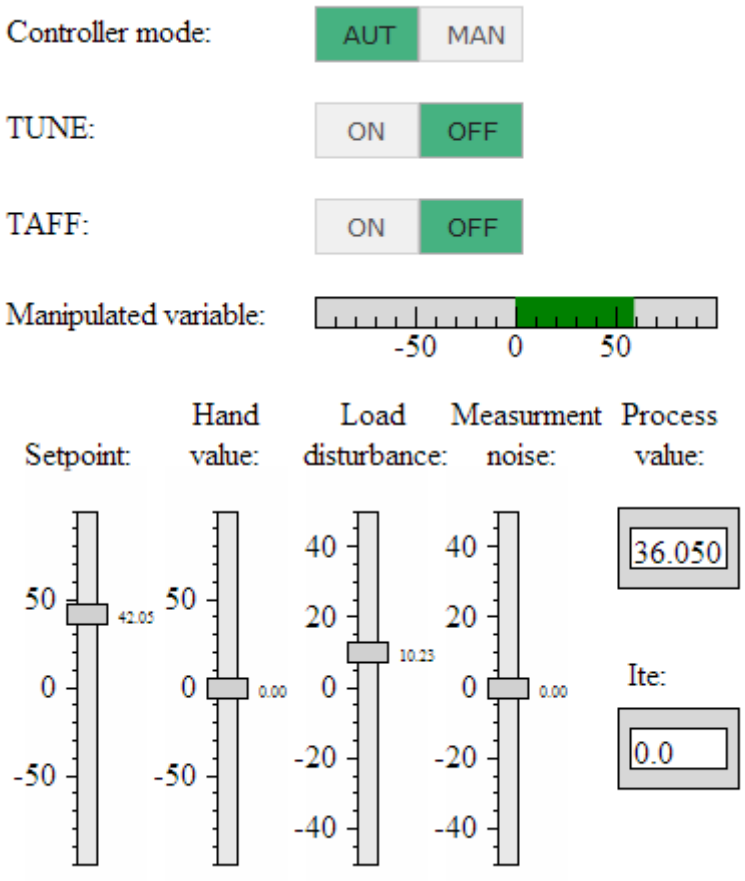
Blok PIDMA je v příkladu zapojen do zpětnovazební smyčky s modelem systému druhého řádu, kterému je možno pomocí bloku s názvem ASTAT (blok CNB umožňující zadání logické konstanty [4]) nastavit astatismus. Dále pak lze blokem pojmenovaným MAN (CNB) přepínat mezi automatickým a manuálním režimem regulátoru. V manuálním režimu se PIDMA chová jako PID regulátor s dvěma stupni volnosti [4]. Další možnosti poskytují uživateli dva funkční bloky typu SG [4] představující řízený generátor signálu. První z nich má název TDIS umožňuje uživateli nastavit vstupní poruchu na systém (defaultně je nastaven obdélníkový typ signálu), druhý se jmenuje PDIS a zpřístupňuje uživateli možnost nastavit úroveň šumu měření (defaultně je zvolen náhodný typ signálu).

6.1.2 Využité grafické objekty

V této virtuální laboratoři byly využity všechny grafické objekty, jejichž tvorbou se zabývaly kapitoly 4 a 5. Pro zpřístupnění základních funkcí laboratoře byl vytvořen panel nástrojů (viz obrázek 22), který obsahuje čtyři instance třídy Slider, dvě instance třídy Display, jednu instanci třídy BarGraf a šest tlačítek typu *radio* definovaných v HTML.

Objekty typu Slider umožňují uživateli zadat požadovanou hodnotu (*setpoint*), která je na vstupu bloku PIDMA označena symbolem sp, požadovanou hodnotu výstupu regulátoru v manuálním režimu (*hand value*), která má symbol hv, amplitudu signálu reprezentujícího chybu na vstupu systému (*load disturbance*) a amplitudu signálu reprezentujícího chybu měření (*measurement noise*). Počáteční hodnoty a tím i umístnění jezdců objektů Slider jsou nastaveny při inicializaci dle příslušných parametrů. Objekt BarGraf je v panelu nástrojů využit jako ukazatel akčního zásahu regulátoru (*manipulated variable*). Minimální a maximální hodnota osy objektu BarGraf je nastavena při vytváření instance grafického objektu dle parametrů hilim a lolim dostupných v bloku PIDMA, které udávají maximální a minimální možnou hodnotu akčního zásahu regulátoru. První objekt Display je využit k zobrazení aktuální hodnoty řízené veličiny

(*process value*). Druhý objekt Display zobrazuje výstup regulátoru ite, který popisuje fáze identifikačního experimentu spuštěného vstupem TUNE. Tlačítka typu *radio* umožňují uživateli přepínat mezi automatickým a manuálním režimem regulátoru. Další tlačítka zpřístupňují spuštění identifikačního experimentu (vstup TUNE regulátoru). Tlačítky je rovněž možno povolit či zakázat přepsání parametrů regulátoru (vstup TAFF). Grafický styl tlačítek je převzat z knihovny jQuery.

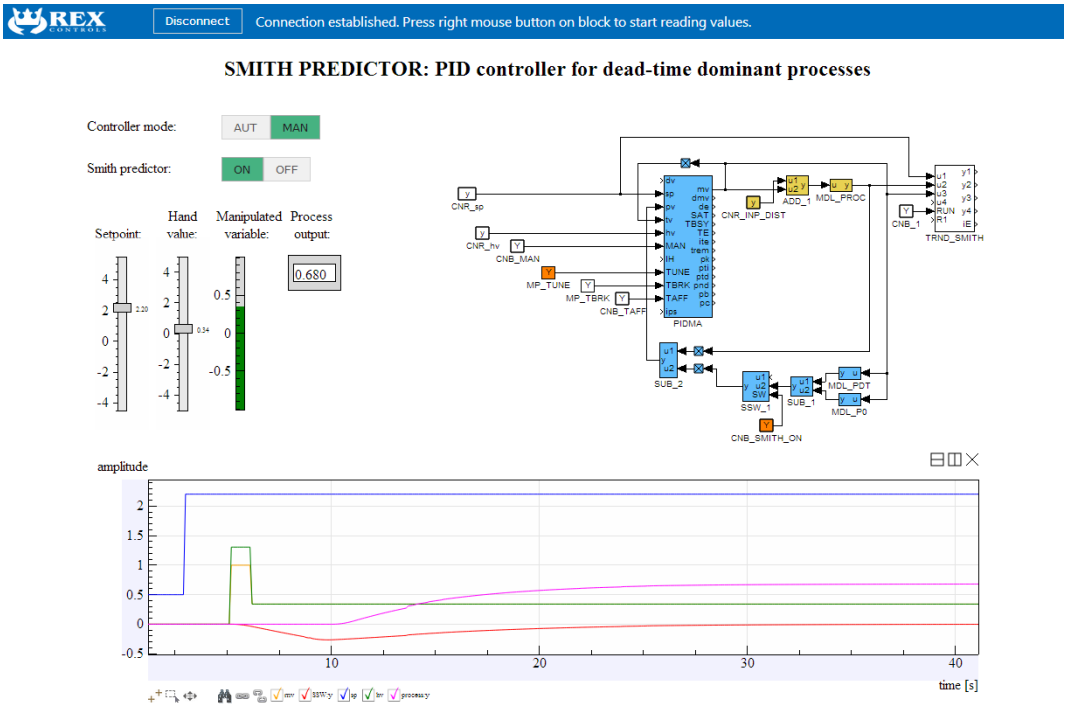


Obrázek 22: Panel nástrojů v příkladu MTUNER

Virtuální laboratoř kromě panelu nástrojů obsahuje také grafický objekt ChartPanel, který obsahuje jednu instanci třídy Trend pro vykreslení časového grafu. Vodorovná osa reprezentuje simulační čas (*time*) v sekundách a svislá osa ukazuje hodnotu amplitudy (*amplitude*) příslušného signálu. Do grafu jsou vkládány aktuální hodnoty čtyř vybraných signálů, které jsou instancemi třídy TrendPen. Zobrazovány jsou požadované hodnoty řízené veličiny (sp), požadované hodnoty výstupu regulátoru v manuálním režimu (hv), akčního zásahu regulátoru (mv) a hodnoty řízené veličiny (pv).

6.2 Smithův prediktor

Virtuální laboratoř demonstrující princip tzv. Smithova prediktoru [3] pro ladění PID regulátoru pro systémy s dominantním dopravním zpožděním (*PID controller for dead-time dominant processes*) je opět založena na schématu převzatém z demonstračních příkladů, které jsou součástí instalace řídicího systému REX.



Obrázek 23: Virtuální laboratoř demonstrující použití Smithova prediktoru

6.2.1 Popis příkladu

Úloha Smithova prediktoru se zabývá použitím PID regulátoru pro systémy s dominantním dopravním zpožděním. Vznik dopravního zpoždění je typický pro kontinuální technologické procesy, ve kterých je přesouván zpracovávaný materiál z místa, kde se nachází akční členy na místo, kde se nachází čidlo snímající regulovanou veličinu. Příkladem takového procesu je regulace tloušťky plechu na válcovací stolici.

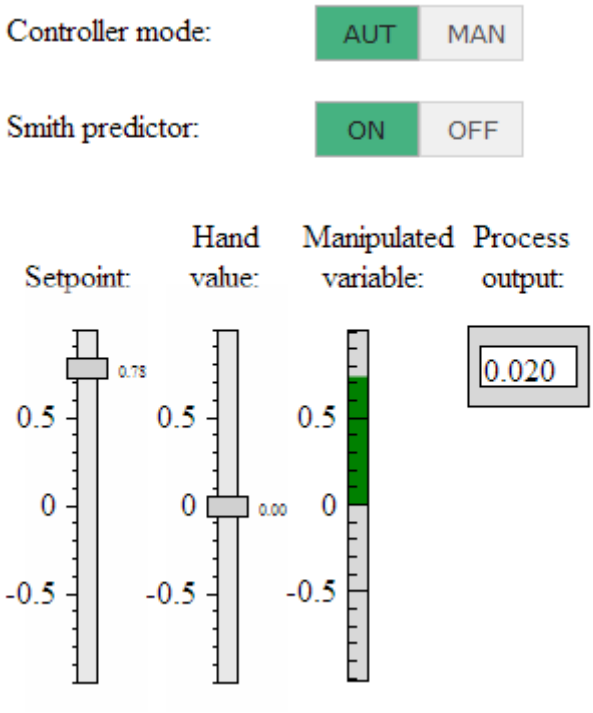
Smithův prediktor má oproti klasické zpětnovazební smyčce ve zpětnovazební větvi navíc model systému s dopravním zpožděním, který je ve schématu (obrázek 23) pojmenován MDL_PDT (blok typu MDL z knihovny RexLib [4]), a model systému bez dopravního zpoždění, ten je ve schématu pojmenován MDL_P0 a jedná se opět o blok typu MDL. Princip spočívá v tom, že výstup modelu bez dopravního zpoždění je odečten od výstupu modelu (blokem SUB_1 typu SUB pro odečet dvou signálů) s dopravním zpožděním a tím je získáno čisté dopravní zpoždění, které je odečteno (blokem SUB_2 typu SUB) od výstupu řízeného procesu. Řízený proces (jeho model) je ve schématu pojmenován MDL_PROC. Tímto způsobem je tedy možné předpovídat budoucí stav sys-

tému neboli je získána informace o regulované veličině bez dopravního zpoždění. Smithův prediktor je aktivní pouze, když má parametr CNB_SMITH_ON (blok CNB) hodnotu ON. V tomto příkladu je použit PID regulátor s momentovým autotunerem (PIDMA), který je možné nastavit způsobem, který byl popsán v předchozím příkladu (6.1.1).

6.2.2 Využité grafické objekty

I v této virtuální laboratoři byly využity všechny grafické objekty, jejichž tvorbou se zabývaly kapitoly 4 a 5. Byl vytvořen panel nástrojů (obrázek 23) zpřístupňující funkce virtuální laboratoře. Panel nástrojů obsahuje dvě instance třídy Slider, jednu instanci třídy Display, jednu instanci třídy BarGraf a čtyři tlačítka typu *radio* definovaná v HTML.

První objekt typu Slider umožňuje zadávání požadované hodnoty (*setpoint*), která je na vstupu bloku PIDMA označena symbolem sp. Druhý objekt typu Slider zpřístupňuje zadávání požadované hodnoty výstupu regulátoru v manuálním režimu (*hand value*), která má symbol hv. Počáteční hodnoty a tím i umístnění jezdců objektů Slider jsou nastaveny při inicializaci dle parametrů načtených ze spuštěné exekutivy řídicího systému REX pomocí serveru webových soketů. Objekt typu BarGraf slouží jako ukazatel akčního zásahu regulátoru (*manipulated variable*). Jeho parametr orientation je (stejně jako u výše zmíněných objektů typu Slider) nastaven na hodnotu *vertical*, což znamená, že je využita vertikální verze obou objektů. Minimální a maximální hodnota osy objektu BarGraf je nastavena při vytváření instance grafického objektu podle hodnot parametrů hilim a lorim udávajících maximální a minimální povolenou hodnotu akčního zásahu regulátoru. Objekt Display zobrazuje hodnoty řízené veličiny (*process value*). První sada tlačítek typu *radio* umožňují uživateli přepínat mezi automatickým a manuálním režimem regulátoru. Druhá sada tlačítek zpřístupňuje zapnutí a vypnutí Smithova prediktoru. Tlačítka jsou standardně součástí HTML. K dosažení jejich grafického stylu je využita knihovna jQuery.



Obrázek 24: Panel nástrojů v příkladu použití Smithova prediktoru

Pod panelem nástrojů se na obrázku 23 nachází grafický objekt ChartPanel obsahující jednu instanci třídy Trend. Na vodorovné ose je simulační čas (*time*) v sekundách a na svislé ose je amplituda (*amplitude*) signálu. Grafický objekt osahuje pět instancí třídy TrendPen, které slouží k vykreslení aktuální požadovaných hodnot řízené veličiny (sp), výstupu regulátoru v manuálním režimu (hv), akčního zásahu regulátoru (mv), řízené veličiny (process:y) a dopravního zpoždění (SSW:y), jež je při zapnutém Smithovo prediktoru odečítáno od výstupu procesu.

7 Závěr

Cílem této práce bylo vytvoření úloh vzdálených a virtuálních laboratoří za použití internetové technologie HTML5. V úvodu byl představen budoucí standard HTML5, programovací jazyk JavaScript, formát SVG a řídicí systém REX. Bylo požadováno pomocí těchto technologií vytvořit komponenty, z nichž budou postaveny virtuální laboratoře. Největší část této práce byla věnována vzniklému grafickému objektu ChartPanel, jehož třídy AxesPanel a Trend zprostředkovávají uživateli nástroje pro práci se zobrazovanými daty v obecném respektive časovém grafu. Dalšími komponentami, které byly vytvořeny a zařazeny do úloh virtuálních laboratoří, jsou grafické objekty Slider (posuvník), Bargraf (sloupcový graf) a Display (displej).

Úvodní kapitola byla věnována rozboru možností internetové technologie HTML5, zejména pak byl ponechán dostatečný prostor pro objasnění nových možností HTML5 oproti starším specifikacím HTML. Dále byl diskutován programovací jazyk JavaScript, jeho vývoj, vzrůstající důležitost a byly zmíněny důvody, proč byly vytvořené komponenty napsány v tomto Jazyce. Zbylá část tohoto oddílu se zabývala seznámením se s formátem SVG, problematikou tzv. webových soketů a řídicím systémem REX.

V následující kapitole byly popsány SCADA a HMI systémy a základní prvky vizualizací. Dále byly definovány požadavky na vizualizaci vytvořenou pomocí technologie HTML5 a byly vybrány komponenty, jejichž tvorbou se tato práce dále zabývala.

Další kapitola byla věnována návrhu architektury a popisu jednotlivých tříd, z nichž je složen grafický objekt ChartPanel, který je napsaný v jazyce JavaScript a využívá možnosti implementace vektorové grafiky do prostředí webového prohlížeče. U každé třídy grafického objektu byly popsány její nastavitelné parametry a funkce, jejichž principy byly demonstrovány na příkladech. Závěr kapitoly obsahuje komentovaný návod, jak uvést grafický objekt ve webové stránce do provozu.

V další kapitole byla popsána a vysvětlena funkčnost ostatních grafických objektů, vytvořených v rámci této práce. Jsou jimi objekt Slider (posuvník), BarGraf (sloupcový graf) a Display (displej). Vytvořené grafické objekty se staly součástí knihoven vizualizačních nástrojů REX.UI a REX.UI.CHARTS.

Poslední kapitola využívá vytvořené komponenty pro tvorbu vzdálených a virtuálních laboratoří. V rámci této práce byly vytvořeny dvě virtuální laboratoře. První z nich demonstruje možnosti nastavení PID regulátoru s momentovým autotunerem. Druhá laboratoř se zabývá tzv. Smithovým prediktorem, tedy použitím PID regulátoru pro řízení systémů s dominantním dopravním zpožděním. Knihovny grafických objektů vytvořily základ pro možnost sestavení dalších laboratoří, které mohou být analogické k těm, prezentovaným v rámci této práce.

Z předchozích odstavců vyplývá, že cíle práce byly splněny. Navíc velkým přínosem je i to, že vytvořené komponenty byly již v průběhu dokončování této práce nasazovány

do praxe a do budoucna je plánován i jejich následný rozvoj a implementace do dalších vzdálených a virtuálních laboratoří.

Literatura

[1] *JavaScript: The Definitive Guide, 6th Edition.* O'Reilly Media, Inc, USA, 2011.

[2] *JavaScript: The Good Parts.* O'Reilly Media, Inc, USA, 2008.

[3] *Lineární systémy 2.* Melichar J., ZČU, Česká republika, 2008

[4] *Funkční bloky systému REX.* Rex Controls, 2012.

[5] *W3C: Specifikace HTML 5* [online], [cit. 8.4.2013]. Dostupné z: <http://www.w3.org/TR/html5>.

[6] *W3C: Specifikace CSS* [online], [cit. 4.5.2013]. Dostupné z: <http://www.w3.org/TR/CSS2/>.

[7] *W3C: Specifikace HTTP* [online], [cit. 4.5.2013]. Dostupné z: < http://www.w3.org/Protocols/rfc2616/rfc2616.html>.

[8] *W3C: Specifikace DOM* [online], [cit. 4.5.2013]. Dostupné z: < http://www.w3.org/TR/domcore/>.

[9] *W3C: Specifikace XML* [online], [cit. 4.5.2013]. Dostupné z: < http://www.w3.org/TR/REC-xml/>.

[10] *W3C: Specifikace SVG* [online], [cit. 4.5.2013]. Dostupné z: < http://www.w3.org/TR/SVG/>.

[11] *W3Schools: SVG tutorial* [online], [cit. 8.4.2013]. Dostupné z: <http://www.w3schools.com/svg/default.asp >.

[12] *W3Schools: JavaScript tutorial* [online], [cit. 8.4.2013]. Dostupné z: <http://www.w3schools.com/js/default.asp >.

[13] *W3Schools: AJAX tutorial* [online], [cit. 4.5.2013]. Dostupné z: < http://www.w3schools.com/ajax/default.asp>.

[14] *W3Schools: jQuery tutorial* [online], [cit. 7.5.2013]. Dostupné z: < http://www.w3schools.com/jquery/default.asp>.

[15] *IBM: Reverse Ajax, Part 1: Introduction to Comet* [online], [cit. 4.5.2013]. Dostupné z: < http://www.ibm.com/developerworks/web/library/wa-reverseajax1/>.

[16] *Web Socket* [online], [cit. 8.4.2013]. Dostupné z: <http://www.websocket.org >.