Západočeská univerzita v Plzni

FAKULTA PEDAGOGICKÁ
KATEDRA VÝPOČETNÍ A DIDAKTICKÉ TECHNIKY

Distanční kurz pro programování v Delphi

DIPLOMOVÁ PRÁCE

Bc. Vojtěch Marton

Učitelství pro střední školy, obor Učitelství informatiky pro střední školy a Učitelství geografie pro střední školy

Vedoucí práce: doc. Ing. Václav Vrbík, CSc.

Plzeň, 2016

Prohlašuji, že jsem diplomovou práci vypracoval samostatně s použitím uvedené literatury a zdrojů informací.

V Plzni, 29. června 2016

.....

vlastnoruční podpis

Poděkování

Děkuji vedoucímu práce, panu doc. Ing. Václavovi Vrbíkovi, CSc, za odbornou pomoc a za další věcné rady a připomínky, při vytváření mé diplomové práce.

ZDE SE NACHÁZÍ ORIGINÁL ZADÁNÍ KVALIFIKAČNÍ PRÁCE.

ZDE SE NACHÁZÍ ORIGINÁL ROZHODNUTÍ O PRODLOUŽENÍ TERMÍNU ODEVZDÁNÍ

OBSAH

ÚVOD

Pokud se řekne slovo programování, většina lidí si představí práci s tisíci řádky zdrojového kódu, ze kterého vzejde výsledný program. Ve školách je pro studenty programování jedním těch nejméně oblíbených předmětů, na které mohou ve škole narazit. Co když tomu tak být nemusí? Co kdyby se programování změnilo v předmět, který bude studenta bavit a kdy student bude sám chtít experimentovat a zkoušet nové věci? O to bych se chtěl pokusit v této diplomové práci, která se zabývá programováním ve vývojovém prostředí Embarcadero Delphi, konkrétně v projektech VCL Forms Application. Jedná se o vývojové prostředí, ve kterém se vytváří programy v jazyce Object Pascal společně s grafickým editorem, ve kterém se vytváří vzhled uživatelského rozhraní programu.

Cílem této diplomové práce, je vytvořit distanční kurz pro předmět Programová 2. Tak jako tento předmět, se tento distanční kurz zabývá programováním ve vývojovém prostředí Embarcadero Delphi, a to v projektech VCL Forms Application. V tomto distančním kurzu se student naučí vytvářet své programy za pomoci grafického editoru a zdrojového kódu. Student bude vytvářet vzhled svých programů pomocí formulářů a komponent a naučí se pracovat s jejich vlastnostmi a událostmi. Následně s těmito komponenty bude umět pracovat i ve zdrojovém kódu. Předpokládá se, že do tohoto kurzu student vstupuje po absolvování předmětu Programování 1, který se zabývá programováním konzolových aplikací za pomoci programovacího jazyka Object Pascal. Student tedy do kurzu vstupuje se znalostmi práce se zdrojovým kódem v jazyce Object Pascal. V kurzu se na tento předmět navazuje kapitolou o objektově orientovaném programování, kdy si student připomene podstatu OOP.

Cílem této práce tedy je, vytvořit studentovi oporu pro studium předmětu Programování 2, a to právě tímto distančním kurzem.

1 CÍLE KURZU

Cílem tohoto kurzu, je seznámit studenty se základními principy práce ve vývojovém prostředí Embarcadero Delphi, konkrétně v projektech VCL Forms Application. Studenti se v tomto kurzu naučí vytvářet vzhled svých aplikací, za pomoci formulářů a komponent. Pochopí principy práce s komponenty, formuláři a s jejich vlastnostmi a událostmi. Studenti se seznámí se základními komponenty. Ke konci kurzu se studenti naučí používat ve svých programech soubory, do kterých si ukládají data z vlastních programů. Student si tímto kurzem rozšíří své logické myšlení, které využije v dalším vzdělávání, nebo při tvorbě vlastních programů. Posledním cílem kurzu je, že po skončení kurzu bude student schopen se dále samostatně rozvíjet v této oblasti a bude plně chápat podstatu objektově orientovaného programování a základy práce v Delphi.

Pro práci s tímto kurzem je třeba, aby student měl na svém počítači nainstalované vývojové prostředí Delphi, protože je žádoucí, aby student své znalosti a dovednosti postavil na praktických cvičeních. Student může pracovat i ve starších verzích tohoto vývojového prostředí, jelikož principy práce jsou stejné.

2 POUŽITÝ SOFTWARE

Tento kurz byl vytvořen v autorském prostředí ProAuthor, který umožňuje vytvářet výukové kurzy za pomoci studijních článků, cvičení, úkolů, testů, autotestů, diskuzí, nebo anket. Obrazové ukázky vývojového prostředí, zdrojových kódů nebo vzhled programů byly pořizovány z vývojového prostředí Embarcadero Delphi 10 Seattle, které je dostupné na webových stránkách Embarcadero v trial verzi na dobu třiceti dnů. Stejně tak v této verzi Delphi byly vytvářeny vlastní příklady, které jsou prezentovány v kurzu jako hotová řešení. V kurzu jsou i obrázky, pomáhající k pochopení principů objektově orientovaného programování. Ty byly vytvářeny v grafickém programu Inscape, který je k dispozici zdarma na webových stránkách Inscape. Poslední použitý program byl určen pro tvorbu animací. Jednalo se o program Captivate 9 od firmy Adobe. Tento program je stejně jako Embarcadero Delphi k dispozici v trial verzi na dobu třiceti dnů.

Jak již bylo v úvodu vysvětleno, Embarcadero Delphi je grafické vývojové prostředí, které je určeno pro tvorbu aplikací v programovacím jazyce Object Pascal. Pro tento kurz jsem si zvolil verzi Delphi 10 Seattle, nazývanou také jako Delphi DX. Tato verze vyšla v polovině roku 2015 a rozšiřuje tak předešlou verzi Delphi XE8. Bohužel Delphi 10 Seattle není nejnovější verzí. Nejnovější verzí pro rok 2016 je Embarcadero Delphi 10.1 Berlin, která rozšiřuje předchozí verzi především o novinky pro vývoj mobilních aplikací. Tuto verzi jsem nepoužil, jelikož tato verze Delphi přináší odlišnou instalaci od předchozích verzí. Při instalaci Delphi 10.1 Berlin se nainstaluje pouze základ vývojového prostředí. Poté je třeba doinstalovat požadované platformy, pro které chceme vyvíjet své aplikace. Bohužel se mi tímto způsobem nepodařilo vývojové prostředí zprovoznit. Proto jsem zvolil verzi předposlední, a to Delphi 10 Seattle.

Již od verze Embarcadero Delphi XE2, lze pomocí vývojového prostředí Delphi vytvářet jak desktopové aplikace, tak aplikace pro mobilní zařízení. To umožňuje výběr vytvářet programy pro operační systémy Windows, Mac OSX, iOS a Android. To znamená, že v Delphi 10 Seattle je mnoho možností, avšak tento kurz se bude především zabývat vytvářením aplikací pro Windows, a to v projektech VCL Forms Application.

3 STRUKTURA DISTANČNÍHO KURZU

Kurz by se dal rozdělit na dvě části. První část je teoretická, která pojednává o objektově orientovaném programování a o vývojovém prostředí Embarcadero Delphi. Druhá část kurzu je zaměřena především na praktické dovednosti, při které se předpokládá, že student bude zároveň s kurzem pracovat i ve vývojovém prostředí Delphi.

Kurz je složen především ze studijních článků a cvičení. Tyto články a cvičení jsou doplněna o úkoly, autotesty a závěrečný teoretický test.

Studijní články jsou zaměřeny pro předání nových informací. Tyto studijní články jsou vždy rozděleny na dvě části, a to na výkladovou část (vlevo) a demonstrační část (vpravo). Ve výkladové části jsou předávané informace studentovi v textové podobě. Text obsahuje i číselné odkazy na obrázek nebo animaci, která se k dané části vztahuje. Demonstrační část obsahuje obrázky nebo animace. Obrázky představují například vzhled a části vývojového prostředí, obrázky představující vlastnosti OOP, vzhled aplikací, anebo části zdrojových kódů. Animace slouží pro předvedení principů práce, nebo funkčnosti programů. Tyto animace obsahují popisky práce, dají se pozastavovat, nebo krokovat. Byla snaha i o to, aby se v demonstrační části studijních článků dali spouštět zdrojové kódy, tak jak je to možné k vidění u distančního kurzu KVD/PGMAP. Z autorovy strany bylo usouzeno, že toto řešení není možné. Bylo to z důvodu, že podstata tohoto kurzu je tvorba programů za použití grafického editoru a zdrojového kódu. Student vytváří vzhled programu za pomoci komponent a formulářů a zároveň pracuje se zdrojovým kódem, ve kterém se pracuje s událostmi a vlastnostmi komponent. Bylo by tedy za potřebí v demonstrační části kurzu spouštět nějaké zjednodušené vývojové prostředí, ve kterém by se pracovalo jak se vzhledem, tak zdrojovým kódem. Pokud bychom chtěli spouštět pouze zdrojový kód, zanedbávali bychom tak z poloviny celý význam práce v projektech VCL Forms Application. Toto by se hodilo spíše pro programování konzolových aplikací, jako v předmětu Programování 1. Vyžadovalo by to také kompilaci programu ze strany serveru a následné stahování hotových programů. Došel jsem tedy k závěru, že jednodušší a lepší varianta bude, když student bude zároveň s kurzem pracovat ve vývojovém prostředí Delphi, které si nainstaluje do svého počítače.

V praktické části kurzu, je ve velké míře využito cvičení. Tato cvičení mají za úkol zadat studentovi práci, na které by měl pracovat samostatně. Zadáním úkolu je vždy program, který má student vytvořit. Toto zadání je podpořeno obrázkem vzhledu požadovaného programu a animací, která znázorňuje funkci programu. Tyto animace jsou stejně jako u studijních článků podpořeny popisky, jdou zastavovat a krokovat. Cvičení také může obsahovat typy pro řešení, které má pomoci studenta nasměrovat správným směrem, při řešení zadání. Nakonec cvičení, je návrh řešení, ve kterém je popsáno možné řešení úkolu. Jelikož jde o programování, princip řešení se může v některých případech lišit od řešení studenta. Nicméně návrh řešení je především určen pro studenty, kteří úkol nebudou schopni vypracovat. Na závěr každého cvičení, je dispozici soubor s hotovým řešením, které lze stáhnout. Stáhne se soubor s vytvořeným projektem, který student otevře v každém vývojovém prostředí Delphi.

V kurzu nalezneme i tři úkoly. Úkoly jsou cvičení velmi podobné. Rozdíl mezi cvičením a úkolem spočívá v tom, že u úkolu není pro studenta viditelné řešení. Úkol slouží jako prověření znalostí a schopností studenta. V úkolu jsou i hodnotící kritéria, podle kterých vyučující výslednou práci studenta hodnotí. Zadání úkolů jsou tvořena stejným způsobem jako zadání cvičení. Mimo zadání je tedy také k dispozici obrázek požadovaného vzhledu programu a animace představující funkčnost programu. K dispozici jsou i typy pro řešení úkolu a nakonec hodnotící kritéria. Řešení je přístupno pro vyučujícího, ve kterém nalezne soubor s hotovým řešením, který lze stáhnout.

V kurzu jsou i dva autotesty, které nalezneme v teoretické části. Tyto auto testy prověřují, zda student dobře pochopil princi objektově orientovaného programování nebo práce ve vývojovém prostředí Delphi. Po odevzdání se test ihned vyhodnotí. Tím dostane student zpětnou vazbu, jak úspěšný byl.

Na závěr kurzu nalezneme závěrečný test. Závěrečný test funguje obdobně jako autotest, jen s tím rozdílem, že u testu je možné nastavit i časový limit.

4 KAPITOLA OBJEKTOVĚ ORIENTOVANÉ PROGRAMOVÁNÍ

První kapitolou distančního kurzu je kapitola Objektově orientované programování. Tato kapitola pojednává o základních principech objektově orientovaného programování a jeho základních rysech. Student zde nalezne i ukázky syntaxe v programovacím jazyce Object Pascal.

Cílem této kapitoly je připomenout studentům v čem spočívá objektově orientované programování. Předpokládá se, že se studenti již s objektově orientovaným programováním setkali, a to minimálně v předmětu Programování 1. Nicméně si myslím, že je důležité si vyjasnit princip objektově orientovaného a jeho základních rysů, aby si studenti případně doplnili mezery ve znalostech. Z tohoto důvodu je tato kapitola do kurzu zařazena. Dalším cílem této kapitoly je, že student pochopí vlastnosti a rysy OOP do takové míry, že bude sám tyto rysy a vlastnosti využívat ve svých programech.

Pro předání učiva studentovi, byly ve studijních článcích zvoleny metody slovní, konkrétně metoda práce s textovým materiálem a názorně demonstrační, konkrétně předvádění, demonstrace obrazů a projekce. Jelikož se jedná o distanční kurz, výkladová část je tvořena textovým materiálem. Tento textový materiál je podpořen demonstrační částí v levé části studijních článku. V této části student nalezne statické obrázky, představující danou problematiku nebo ukázky částí zdrojových kódů. Student se také v demonstrační části setká s projekcí animací, které demonstrují principy práce.

Informace v této kapitole jsou čerpány z publikací Slavoj Písek - Začínáme programovat v Delphi, James Keogh – OOP bez předchozích znalostí a z bakalářské práce Vojtěch Marton – Tvorba sady příkladů pro předmět Programování 2.

V této kapitole nalezneme pět studijních článků a jeden autotest.

4.1 STUDIJNÍ ČLÁNEK: OBJEKTY A TŘÍDY

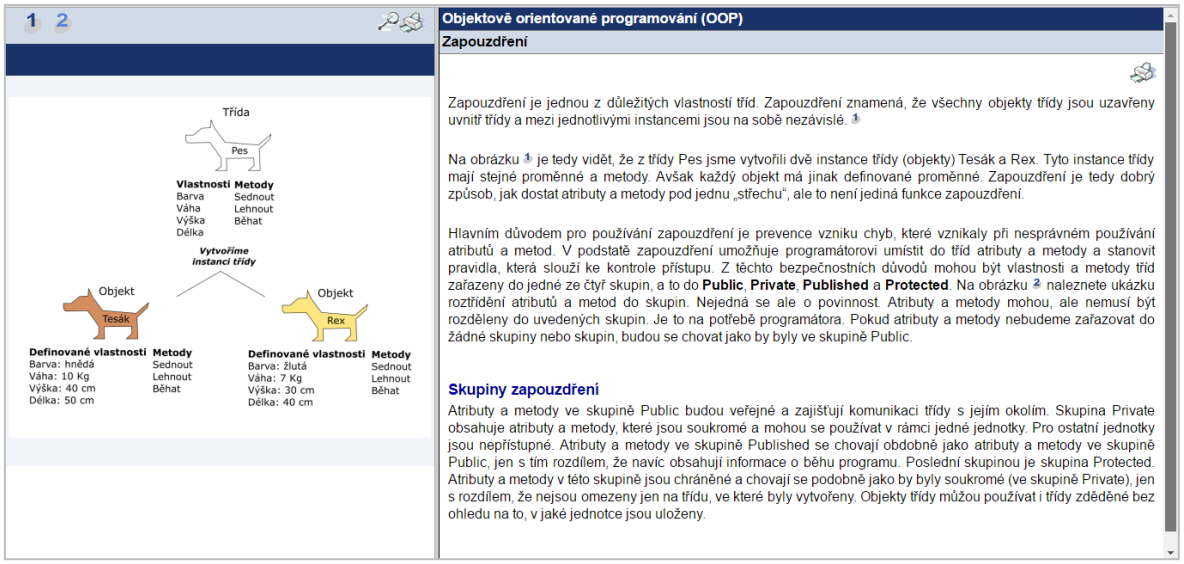
Cílem tohoto studijního článku, je připomenout studentovi co je základní filozofií objektově orientovaného programování, co je to objekt v OOP, jak se vytváří a ruší a jaký je rozdíl mezi třídou a objektem. Problematika objektů je zde popisována tak, že lze objekty chápat jako věci okolo nás, jako například dům, auto apod. Tyto věci mají nějaké vlastnosti a chování. Následně je úkolem programátora za pomoci objektově orientovaného programovacího jazyka tyto objekty přenést do programu za pomoci tříd. Třídy jsou v tomto článku popisovány jako takové šablony objektů. To ilustruje i obrázek v demonstrační části článku. Student si též připomene, z jakého důvodu se objekty musí vytvářet, jak je vytvářet a proč by se měli po použití opět rušit. V demonstrační části článku, jsou ukázky syntaxe v jazyce Object Pascal, pro vytvoření třídy, objektu a zrušení objektu.

Obrázek 1: Studijní článek Objekty a třídy

4.2 STUDIJNÍ ČLÁNEK ZAPOUZDŘENÍ

Tento studijní článek pojednává o jednom z hlavních rysů objektově orientovaného programování, a to zapouzdření. Student se zde obeznámí s důvody zavedení zapouzdření, a jak se zapouzdření v programování projevuje. Zapouzdření je v tomto článku popisováno ze dvou hledisek. Prvním je, že ze třídy lze vytvořit více objektů. Tyto objekty mají stejné atributy a vlastnosti. Nicméně zapouzdření zajišťuje, že vlastnosti a metody těchto objektů budou na sobě nezávislé. Druhým hlediskem je možnost zařazení atributů a metod do skupin, které určují práva k přístupu z okolních objektů. Nezávislost

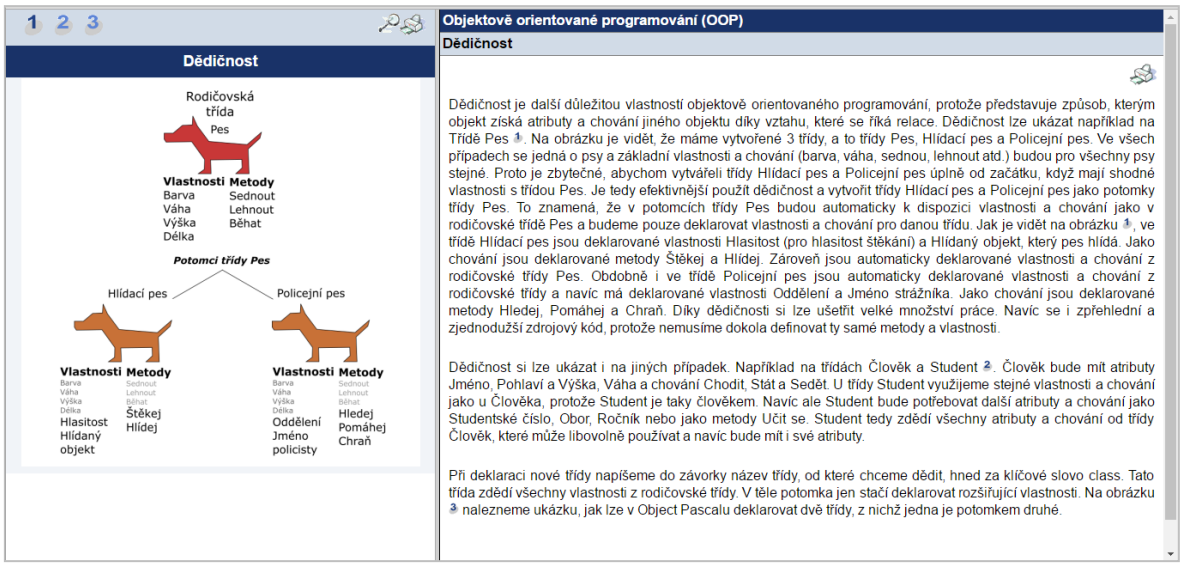
atributů a metod jednoho objektu od druhého, je graficky znázorněn v demonstrační části kurzu. V této části je i ukázka syntaxe v jazyce Object Pascal, jak lze rozdělit atributy a metody do daných skupin.



Obrázek 2: Studijní článek Zapouzdření

4.3 STUDIJNÍ ČLÁNEK DĚDIČNOST

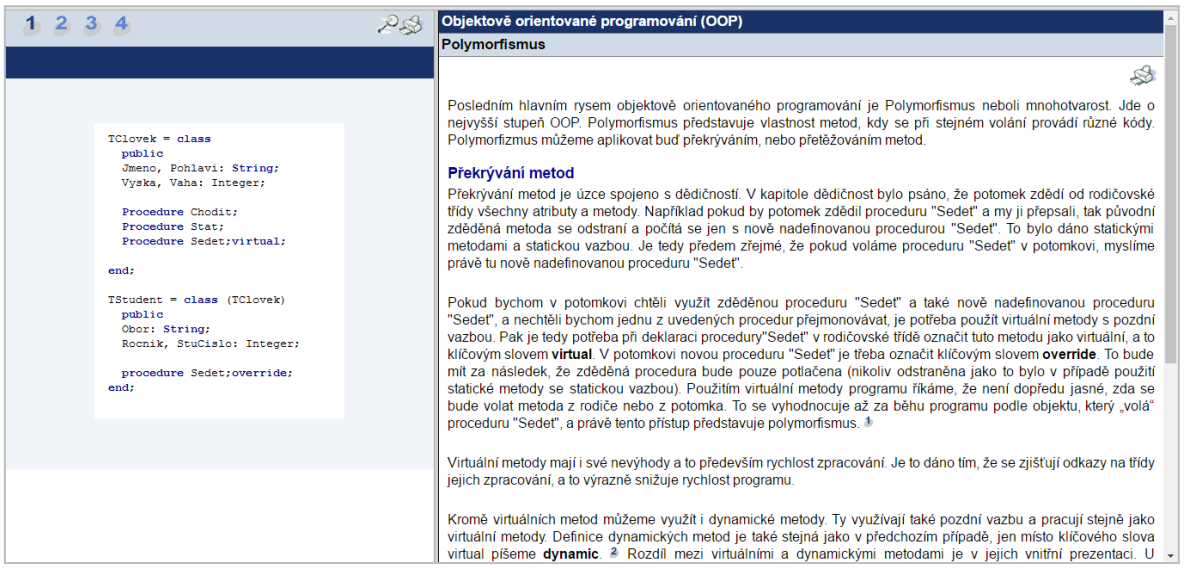
Tento studijní článek pojednává o druhém z hlavních rysů objektově orientovaném programování, a to o dědičnosti. Cílem tohoto studijního článku, je připomenout studentům co to dědičnost je, jak ji lze využít, v čem programátorovi pomáhá a jaká má omezení. Dědičnost je v tomto článku popisována za pomoci příkladu třídy Pes. Z této rodičovské třídy vytvoříme dva potomky, a to třídu Hlídací pes a Policejní pes. Tento příklad je i graficky znázorněn v demonstrační části kurzu. Dědičnost je popsána ještě druhým příkladem, a to na třídě Člověk. Z této vytvoříme potomka, třídu Student. Tento příklad je také znázorněn graficky v demonstrační části článku. Ke konci článku je také vysvětlena syntaxe v jazyku Object Pascal, která je i předvedena v demonstrační části článku.



Obrázek 3: Studijní článek Dědičnost

4.4 STUDIJNÍ ČLÁNEK POLYMORFISMUS

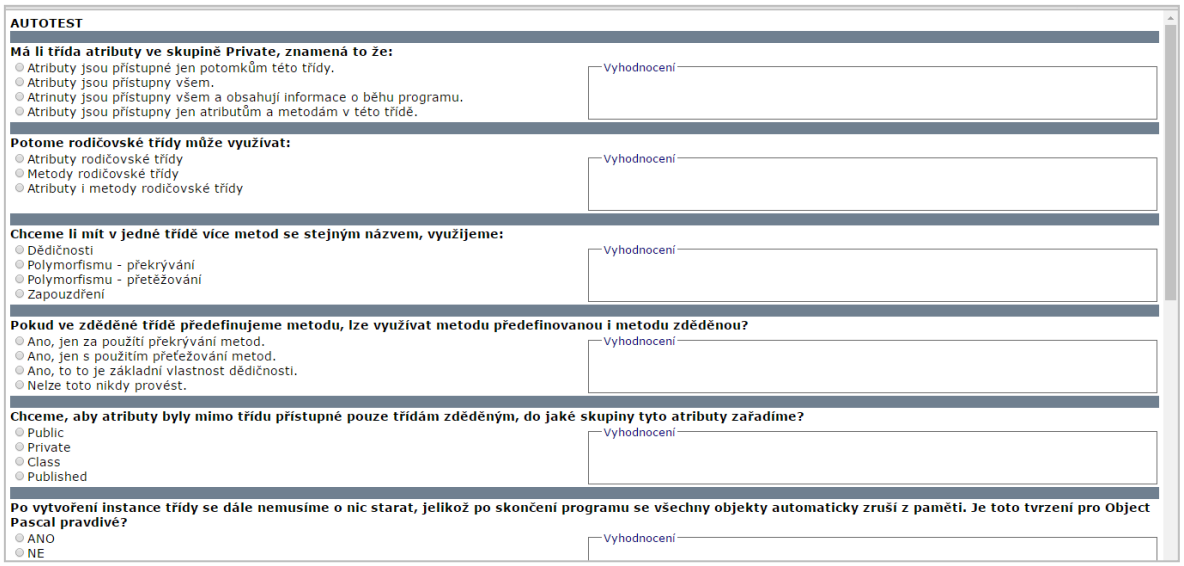
Tento studijní článek pojednává o třetím hlavním rysu objektově orientovaném programování, a to o Polymorfismus. Jedná se o nejvyšší stupeň OOP. Jelikož se jedná o náročnější téma než předešlá témata zapouzdření a dědičnost, je snaha o co nejlepší popis této problematiky. Předpokládá se, že pokud studenti budou mít mezery ve znalostech OOP, tak u tohoto tématu budou mezery největší. Cílem je studenta obeznámit s tím, co to polymorfizmus je, jak se v Object Pascalu projevuje, jaké jsou jeho výhody a nevýhody a kdy je dobré vlastnosti polymorfismu využít. Student se také seznámí se syntaxí v Object Pascalu. Polymorfismus v Object Pascalu je rozdělen na dva druhy, a to na překrývání a přetěžování metod. V části o překrývání metod je popsáno, že v Object Pascalu lze metody překrývat třemi způsoby, a to změnou metody v rodičovské třídě na metodu virtuální, dynamickou nebo abstraktní. Ke každému způsobu, je popis o jeho výhodách a nevýhodách a o způsob zápisu. Ukázky syntaxe studenti opět naleznou v demonstrační části článku. Nakonec je v článku popsán druhý způsob projevu polymorfismu, a to přetěžování metod. Zde je popsáno, jakým způsobem se přetěžování metod projevuje, jak lze metody přetížit a jaké výhody to přináší. Student opět v demonstrační části článku nalezne ukázku zápisu v Object Pascalu.



Obrázek 4: Studijní článek Polymorfismus

4.5 AUTOTEST OOP

Tento autotest si klade za cíl, prověřit studentovu znalost principů objektově orientovaného programování. Zde je připraveno devět otázek, které jsou zaměřeny na hlavní rysy OOP, a to na zapouzdření, dědičnost a polymorfismus. Tento test se skládá z otázek polytomických (jedna správná odpověď) a dichotomických (jedna správná odpověď, ze dvou možných, například Ano-Ne). Po odevzdání autotestu dostane student zpětnou vazbu o správnosti jeho odpovědí. U některých otázek je poskytnuto i vysvětlení, pokud student odpoví na danou otázku chybně.



Obrázek 5: Autotest OOP

5 KAPITOLA VÝVOJOVÉ PROSTŘEDÍ EMBARCADERO DELPHI

Druhou kapitolou tohoto distančního kurzu, je kapitola Vývojové prostředí Embarcadero Delphi. Cílem této kapitoly, je seznámit studenty s vývojovým prostředím Embarcadero Delphi a s významem a funkcí těch nejdůležitějších nástrojů, které bude student využívat při tvorbě svých programů. Student ke konci této kapitoly bude schopen využívat ty nejdůležitější nástroje, které mu umožní vytvářet vzhled a zdrojový kód programu. V této kapitole se student naučí pracovat s nástroji Project Manager, Tool Palette, Object Inspector, Structure, horními panely nástrojů a samotným editorem programu.

Tato kapitola je pro studenta stěžejní, jelikož je třeba, aby dokázal pracovat ve vývojovém prostředí Delphi za pomoci všech uvedených nástrojů. Jelikož se v této části kurzu zabýváme nástroji pro práci, zabíháme tím i k principům práce v projektech VCL Forms Application. Princip práce jako takový si student sám vyzkouší v následující kapitole.

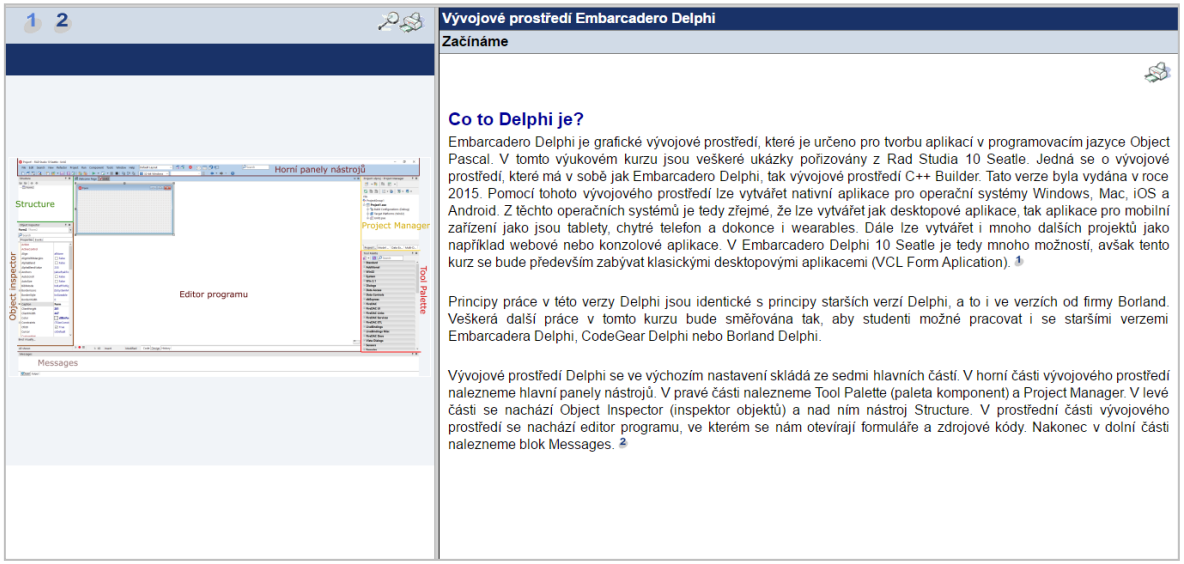
Metody pro předání učiva byly voleny obdobně jako u předchozí kapitoly. Pro předání učiva studentovi, byly ve studijních článcích zvoleny metody slovní, konkrétně metoda práce s textovým materiálem a názorně demonstrační, konkrétně předvádění, demonstrace obrazů a projekce. Jelikož se jedná o distanční kurz, výkladová část je tvořena textovým materiálem. Tento textový materiál je podpořen demonstrační částí v levé části studijních článku. V této části student nalezne statické obrázky, představující danou problematiku nebo ukázky částí zdrojových kódů. Student se také v demonstrační části setká s projekcí animací, které demonstrují principy práce.

Informace v této kapitole jsou čerpány z publikací Slavoj Písek - Začínáme programovat v Delphi a z bakalářské práce Vojtěch Marton – Tvorba sady příkladů pro předmět Programování 2.

5.1 STUDIJNÍ ČLÁNEK ZAČÍNÁME

Tímto studijním článkem se student seznámí s vývojovým prostředím Delphi, při tvorbě projektů VCL Forms Application. Cílem tohoto studijního článku je představit studentovi vývojové prostředí Delphi, vysvětlit co to vlastně Delphi je a s jakými nástroji se setká. Je zde také popsáno, že se student v tomto kurzu bude setkávat s ukázkami z vývojového prostředí Delphi, verze 10 Seattle, ale ať jsou veškeré ukázky v tomto kurzu pořizovány

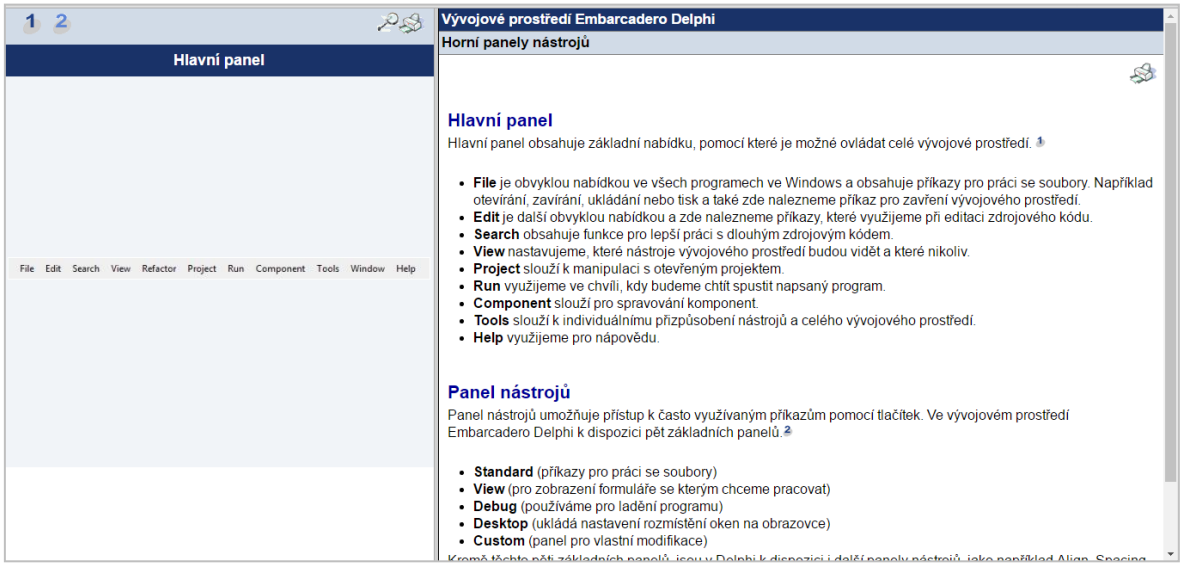
z této verze, principy práce, používané komponenty, nebo práce s vlastnosti a událostmi bude identická i s předchozími verzemi. Také je tato verze stručně představena.



Obrázek 6: Studijní článek Začínáme

5.2 STUDIJNÍ ČLÁNEK HORNÍ PANELY NÁSTROJŮ

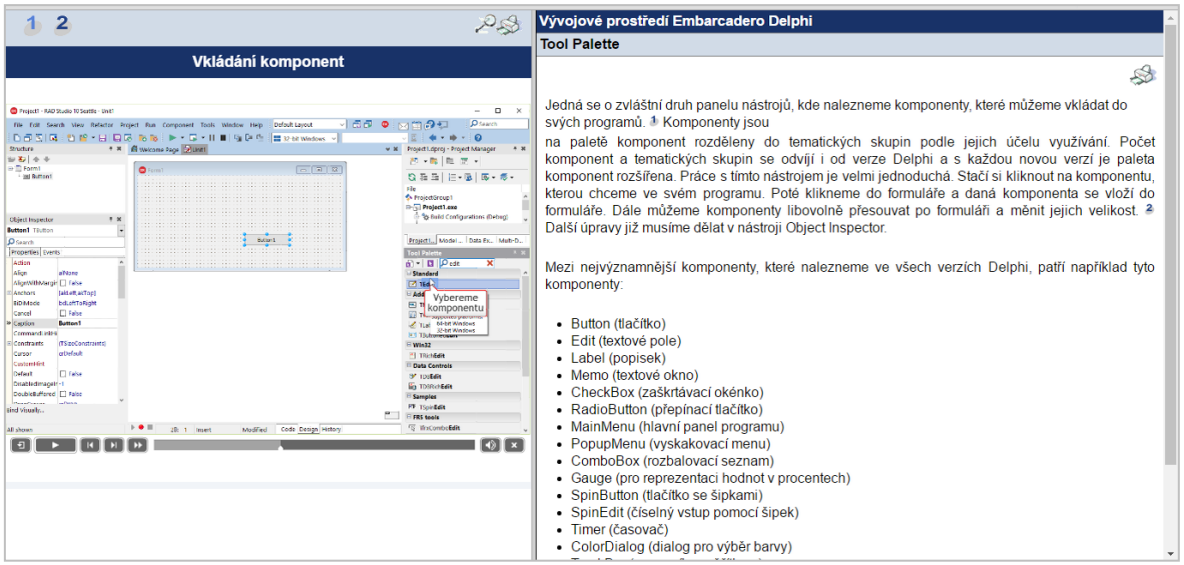
Cílem tohoto studijního článku, je seznámit studenta se základní nabídkou vývojového prostředí Embarcadero Delphi. Tuto nabídku tvoří hlavní panel a panel nástrojů. Student v tomto článku nalezne popis nejdůležitějších částí těchto panelů.



Obrázek 7: Studijní článek Horní panely nástrojů

5.3 STUDIJNÍ ČLÁNEK TOOL PALETTE

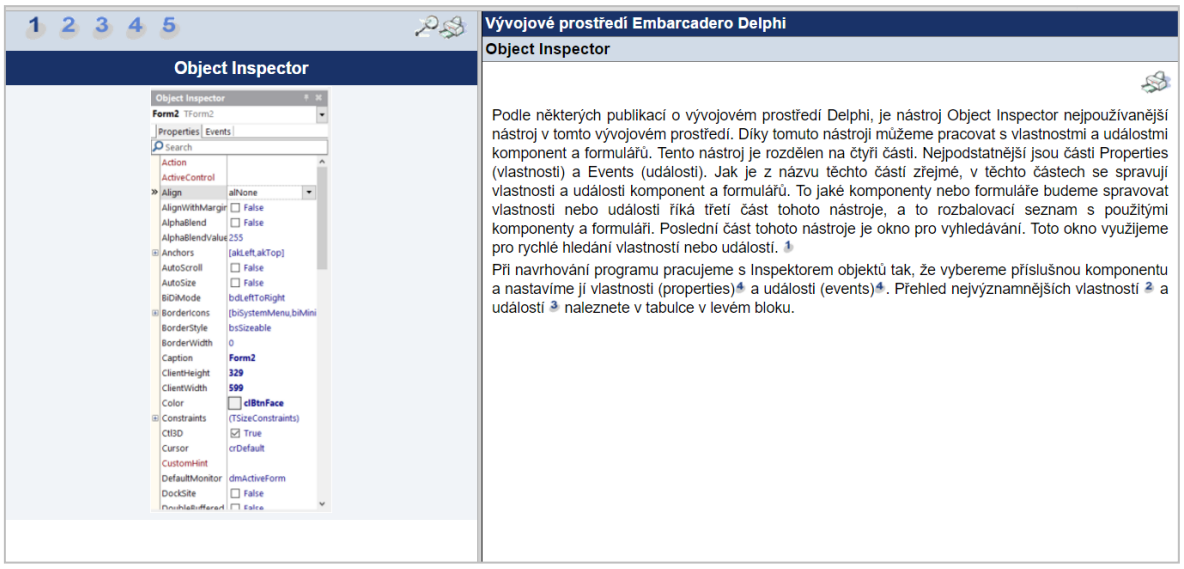
Tento studijní článek se zabývá jedním z nejdůležitějších nástrojů pro tvorbu vzhledu aplikace. Jedná se o nástroj Tool Palette. Cílem tohoto studijního článku je představit studentovi tento nástroj, ve kterém najde komponenty. Student zde najde i animaci, ze které pochopí co to komponenta je, jak jsou řazeny v nástroji Tool Palette, jak lze komponenty vyhledávat podle názvu a jak je vkládat do formuláře. Součástí článku je výpis názvů nejběžnějších komponent s vysvětlením co daná komponenta představuje. Dále se tento článek nezabývá komponenty, jelikož je k tomu určen vlastní studijní článek.



Obrázek 8: Studijní článek Tool Palette

5.4 STUDIJNÍ ČLÁNEK OBJECT INSPECTOR

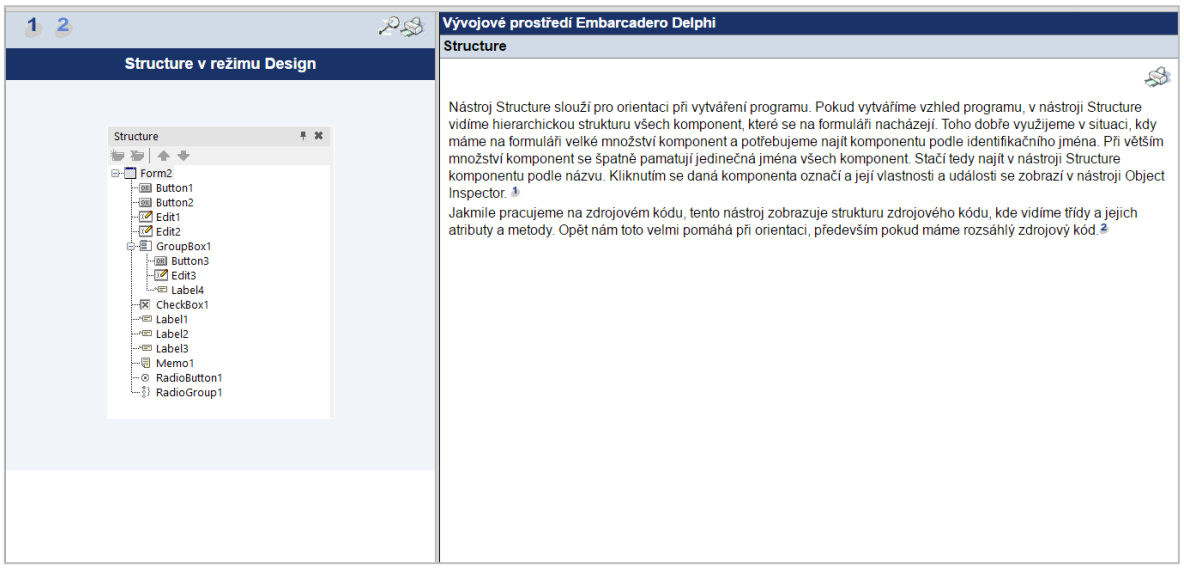
Tento studijní článek pojednává o nástroji Object Inspector, který je pro práci s komponenty velmi důležitý. Cílem tohoto studijního článku, je studenta seznámit s funkcí a významem tohoto nástroje. Student v tomto článku nalezne tabulku s nejčastějšími vlastnostmi a událostí komponent, díky tabulkám, které nalezne v demonstrační části článku. V této části nalezne animace, představující příklad práce s tímto nástrojem. Cílem tohoto článku však není studenta naučit pracovat s vlastnostmi komponent nebo vytvářet události. Jedná se pouze o příklad, který demonstruje funkci nástroje.



Obrázek 9: Studijní článek Object Inspector

5.5 STUDIJNÍ ČLÁNEK STRUCTURE

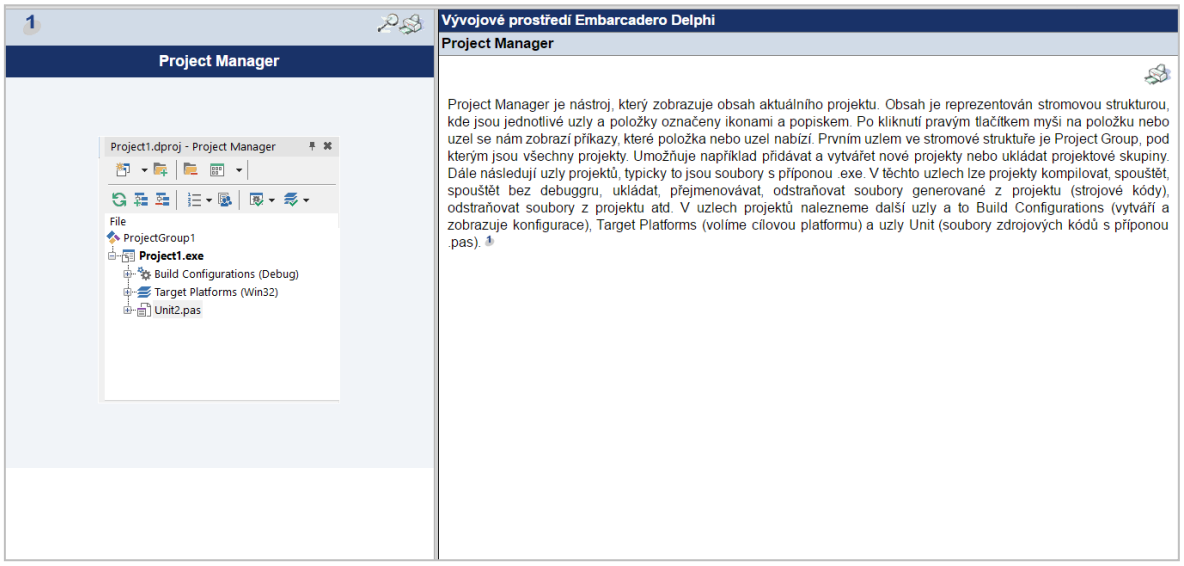
Neméně důležitým nástrojem v Delphi, je nástroj Structure. Cílem tohoto studijního článku je studenta seznámit s tímto nástrojem. Student skutečný význam tohoto nástroje ocení až při praktickém cvičení. Důležité je, aby student byl poučen o tomto nástroji a věděl jak s ním zacházet. V demonstrační části článku jsou dva obrázky, které představují vzhled nástroje Structure při vytváření vzhledu a při práci se zdrojovým kódem.



Obrázek 10: Studijní článek Structure

5.6 STUDIJNÍ ČLÁNEK PROJECT MANAGER

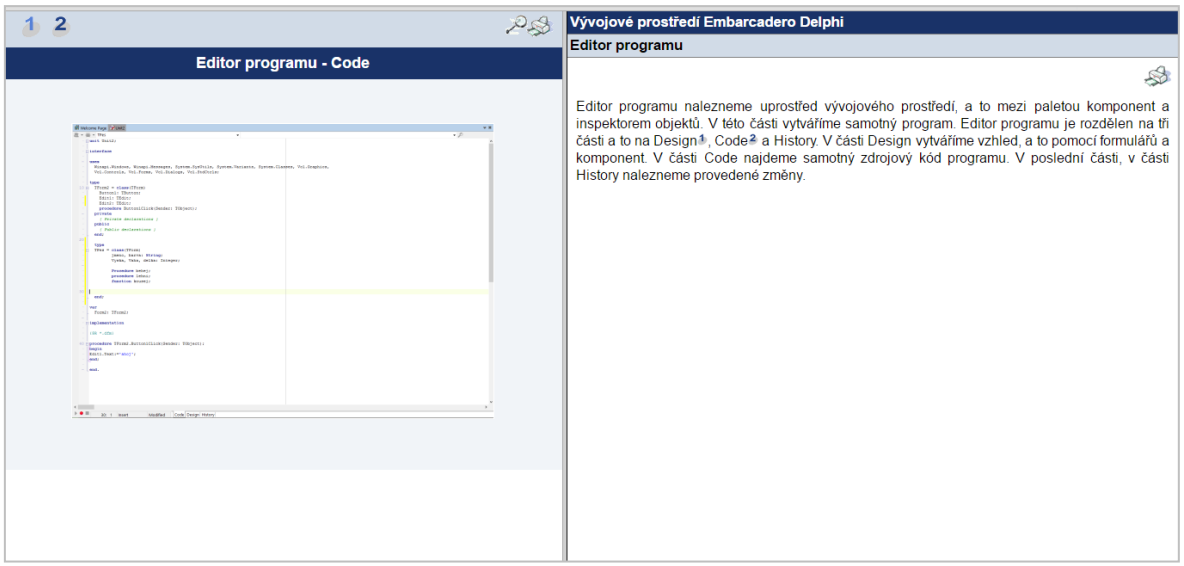
V tomto studijním článku se studenti seznámí s nástrojem Project Manager. Cílem je opět studenty seznámit s tímto nástrojem a s jeho možnostmi. Jeho skutečný význam poznají až při praktickém cvičení. Studenti v demonstrační části naleznou jen jeden obrázek, představující vzhled tohoto nástroje.



Obrázek 11: Studijní článek Project Manager

5.7 STUDIJNÍ ČLÁNEK EDITOR PROGRAMU

Posledním studijním článkem v této kapitole pojednává o Editoru programu. Zde se nejedná o nástroj jako takový, ale o hlavní část vývojového prostředí, ve kterém se odehrává tvorba vzhledu a zdrojového kódu. V této části vývojového prostředí, budou studenti pracovat právě na svých programech. Cílem tohoto článku je studenty seznámit s možnostmi tohoto nástroje. Je důležité, aby studenti věděli jak přepínat mezi editací vzhledu a zdrojového kódu.



Obrázek 12: Studijní článek Editor programu

6 KAPITOLA ZÁKLADY PRÁCE V DELPHI

Touto kapitolou přecházíme pomalu do praktické části kurzu. Studenti se zde prakticky seznámí s vývojovým prostředím Delphi a utvrdí si poznatky z předchozí kapitoly. Od této části kurzu se předpokládá, že student bude pracovat zároveň i ve vývojovém prostředí Delphi, kde si sám vyzkouší danou problematiku. Nyní je stěžejní studentova praktická činnost, protože jedině praxí lze tento kurz úspěšně dokončit.

Hlavním cílem této kapitoly, je seznámit studenty s prací s komponenty, formulářem a s jejich vlastnostmi a událostmi. Dále je cílem naučit studenty vytvářet projekty VCL Forms Application, kompilovat a spouštět projekty (tím si studenti vytvoří první program) a pochopit tak základní principy práce ve vývojovém prostředí Delphi.

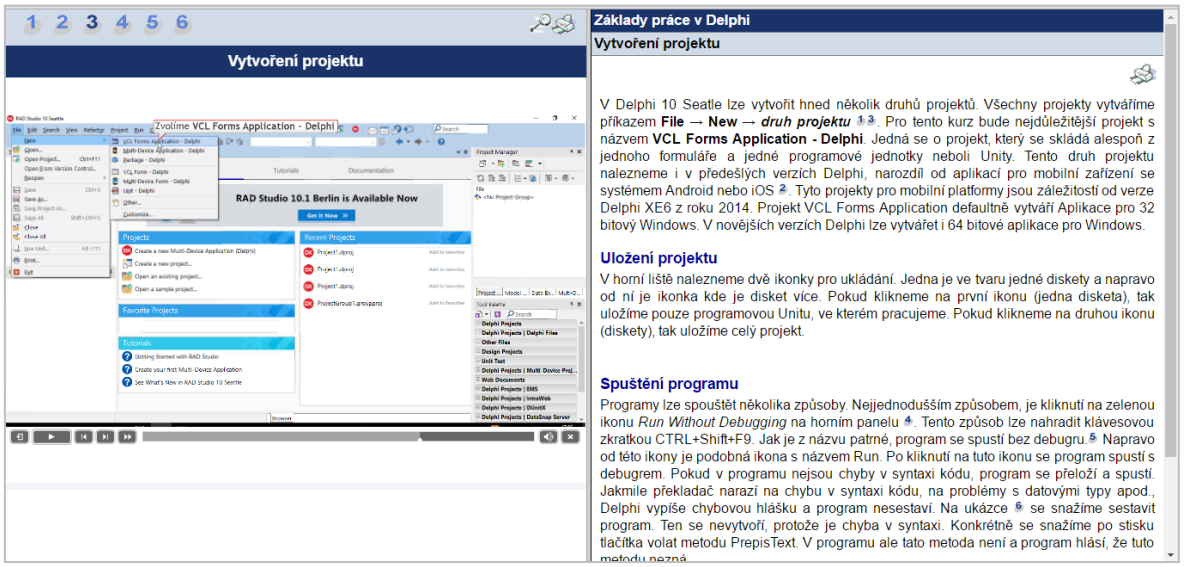
Studenti si prakticky vyzkouší práci s hlavními nástroji, především v Tool Palette, Object Inspector, Structure a s Editorem programů. Dále si prakticky vyzkouší pracovat s komponenty, když je budou hledat v nástroji Tool Palette a také vkládat do formuláře. Poté budou komponenty přesouvat a měnit jejich velikost přímo ve formuláři. Následně se naučí editovat vlastnosti komponent a formuláře a vytvářet události, a to vše pomocí nástroje Object Inspector. Student se v této kapitole seznámí i s prvními komponenty a jejich vlastnostmi. Student se seznámí s prvními komponenty, konkrétně s komponenty Edit, Label a Button. Poprvé budou studenti pracovat i se samotným zdrojovým kódem, ve kterém budou pracovat s vlastnostmi a událostmi komponent.

Ke konci kapitoly jsou připravena dvě praktická cvičení a jeden úkol, na kterých bude student pracovat samostatně. Na závěr kapitoly je připraven teoretický autotest, kterým se prověří znalosti studenta z posledních dvou kapitol.

6.1 STUDIJNÍ ČLÁNEK VYTVOŘENÍ PROJEKTU

Cílem tohoto studijního článku, je naučit studenta vytvořit projekt VCL Forms Application, uložit a také spustit. Konkrétně tento studijní článek pojednává o možnostech vytvoření projektu, jelikož bude-li mít student k dispozici novější verzi vývojového prostředí delphi, bude moc vytvářet i aplikace pro mobilní zařízení. Dále se student dozví jak uložit pouze programovou unitu nebo celý projekt. Na konec této kapitoly se student seznámí, jakým způsobem své programy lze spouštět. Student je i obeznámen se situací, kdy kompilátor objeví chybu v syntaxi.

V demonstrační části je statickým obrazem znázorněná cesta k vytvoření projektu VCL Forms Application. Pro zajímavost zde student nalezne i obrazovou ukázku vytvoření projektu pro mobilní aplikace. Dále tu jsou k dispozici i tří animace, které znázorňují vytvoření projektu, spuštění projektu se správnou a i špatnou syntaxí.



Obrázek 13: Studijní článek Vytvoření projektu

6.2 STUDIJNÍ ČLÁNEK PRÁCE S KOMPONENTY

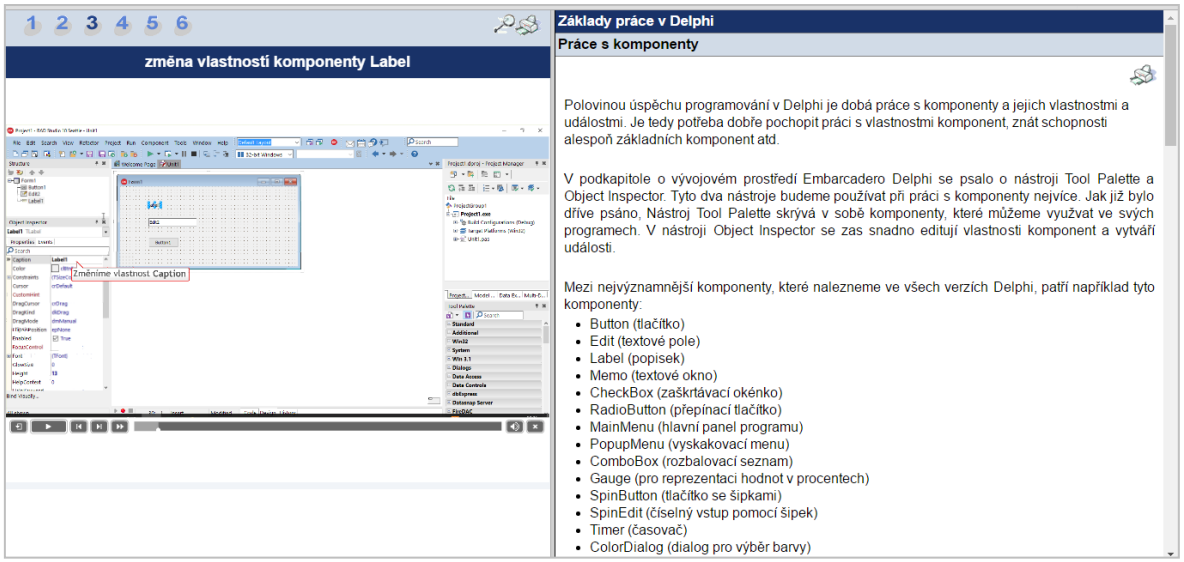
Tento studijní článek pojednává o komponentech, práci s nimi a o práci s jejich vlastnostmi a událostmi. Dalo by se říct, že polovinou úspěchu programování v projektech VCL Forms Application, je dobrá práce s komponenty, především s jejich vlastnostmi a událostmi. Z tohoto důvodu je tento článek stěžejní pro další studentovu práci v tomto kurzu. Cílem tohoto kurzu, je seznámit studenty s nejběžnějšími komponenty, s možností vkládání komponent do formuláře a pracovat s jejich vlastnostmi, a to jak v kódu, tak v nástroji Object Inspector. Nejpodstatnější na tomto článku je ta část, kdy se jedná o vlastnostech, o jejich datových typech a o práci s nimi přímo ve zdrojovém kódu. Pokud student nebude umět pracovat s vlastnostmi komponent přímo ve zdrojovém kódu, či nepochopí problematiku datových typů vlastností, nebude moci dále pracovat na cvičeních a úkolech.

Není cílem studenta v tomto článku práci s komponenty naučit, ale jen seznámit. Problematiku práce v komponenty student zvládne až při praktickém cvičení, kdy si tyto nabyté znalosti osvojí.

Ke konci článku se student seznámí s neméně důležitým tématem, a to s událostmi komponent a formulářů. Student se v této části článku dozví, co to jsou události, s jakými nejběžnější událostmi se lze setkat a jak se vytvářejí.

V demonstrační části článku student nalezne čtyři animace a dva obrázky. První animace představuje práci s komponenty, ve které student uvidí jakými způsoby, lze komponenty do formuláře vkládat. Tato animace byla již ve studijním článku o nástroji Tool Palette, tam však sloužila jen jako příklad použití, nástroje. Druhá a třetí animace představuje ukázku, jak lze měnit vlastnosti komponent Label a Edit v nástroji Object Inspector. K vlastnostem komponent slouží i tabulka nejběžnějších vlastností, která je v kurzu vložena jako obrázek v demonstrační části kurzu. Čtvrtá animace představuje ukázku vytvoření událostí, konkrétně vytvoření události OnClick komponenty Button. V této animaci student uvidí, že na kartě Events v nástroji Object Inspector si stačí vybrat danou událost, poklepat na ni a v kódu se vytvoří procedura pro zapsání těla události.

Předpokládá se, že zároveň s kurzem, bude student pracovat ve vývojovém prostředí, kdy si sám vyzkouší vkládání komponent do formuláře, práci s jejich vlastnostmi v nástroji Object Inspector a vytvoření události.



Obrázek 14: Studijní článek Práce s komponenty

6.3 STUDIJNÍ ČLÁNEK PRVNÍ PROGRAM

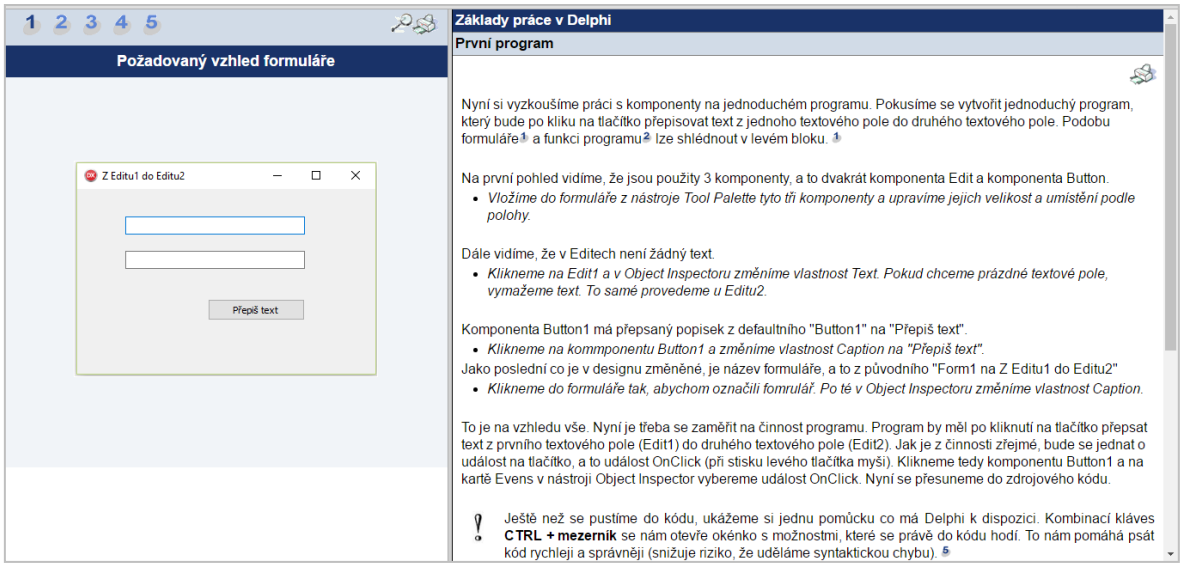
Hlavním cílem tohoto studijního článku, je využít nabyté znalosti z předchozího studijního článku o komponentech a jejich vlastnostech a událostech. Cílem je, aby student tyto znalosti přenesl do praxe, a vytvořil si první program. Nejedná se však o cvičení, ale o podrobný popis práce ve studijním článku. Druhým cílem je, seznámit studenty s možností nápovědy při psaní zdrojového kódu, která programátorovi napovídá s části příkazů. Tím si student ulehčí a urychlí práci. Také se snižuje riziko syntaktické chyby při psaní kódu. V demonstrační části článku je k dispozici animace, představující použití nápovědy.

Praktická ukázka je formou zadání úkolu, který je ve studijním článku řešen krok po kroku. Úkolem je vytvořit jednoduchý program, který po kliknutí na tlačítko přepíše text z jednoho textového pole do textového pole druhého. Tato praktická ukázka je soustředěna jen na práci s komponenty a s jejich vlastnostmi a událostmi. Nyní se nezabýváme třídami, objekty, funkcemi apod. Je třeba, aby si student touto praktickou ukázkou a následujícími cvičeními osvojil práce s komponenty.

V demonstrační části studijního článku vidí student vzhled a funkci programu, který má student vytvořit. Vytvořením vzhledu tohoto programu si student vyzkouší pracovat s komponenty Edit (textové pole) a Button (tlačítko). Vyzkouší práci s jejich vlastnostmi v nástroji Object Inspector, konkrétně u komponenty Button s vlastností Caption a u komponenty Edit s vlastností Text. Také si student změní nápis v hlavičce okna programu, a to změnou vlastní Caption formuláře.

V této chvíli má student vzhled programu hotový, nyní si vytvoří událost OnClick komponenty Button. V této události se bude přepisovat text z prvního textového pole do druhého textového pole, a to po kliku na tlačítko. V této události, si student poprvé vyzkouší pracovat s vlastnostmi komponent přímo ve zdrojovém kódu. Pro řešení tohoto problému jsou dvě možnosti. První je, že text z prvního textového pole rovnou přiřadíme do vlastnosti Text druhého textového pole, anebo obsah prvního textového pole přiřadíme nejdříve do lokální proměnné, kterou poté vložíme do druhého textového pole. První z možností je rychlejší, nic méně pro některé studenty méně přehledná nebo komplikovaná. Druhá možnost je sice zbytečně dlouhá, ale alespoň přehledná, kdy je zřejmé přiřazování hodnot z jedné komponenty do druhé. Obě tyto varianty jsou

ve výkladu popsány a v demonstrační části článku ukázány. Na konci výkladové části je i k dispozici hotové řešení, které si student může stáhnout a projekt otevřít ve svém vývojovém prostředí.



Obrázek 15: Studijní článek První program

6.4 CVIČENÍ SOUČET DVOU ČÍSEL

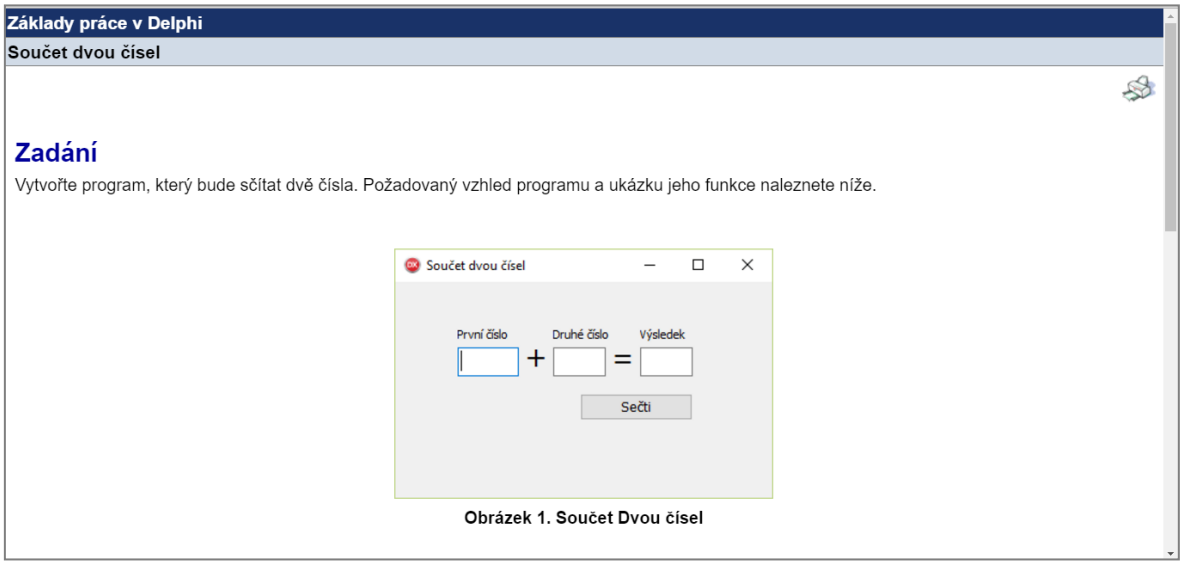
Cílem tohoto cvičení, je osvojení si nabytých znalostí o práci s komponenty. Toto cvičení se opět zabývá pouze prací s komponenty, formulářem, vlastnostmi a událostmi. Student v tomto cvičení má za úkol vytvořit program, který bude sčítat dvě čísla. K dispozici má požadovaný vzhled programu, animaci, která znázorňuje funkci programu a typy pro řešení. Na tomto cvičení by měl student pracovat samostatně. Nepředpokládá se, že student z předešlých studijních článků vše dokonale pochopil, proto se počítá i s vyšší časovou náročností tohoto cvičení. V kurzu je k tomuto cvičení doporučeno 40 minut práce.

Student by neměl příliš váhat při tvorbě vzhledu, jelikož se jedná o již známé komponenty. Při tvorbě vzhledu tohoto programu bude pracovat s komponenty Button Edit a Label. S komponentou Label (popisek) student dosud nepracoval a z tohoto důvodu je v typech pro řešení popsány vlastnosti této komponenty, na které se má student zaměřit. Navíc pokud bude student pozorný, všimne si, že u některých komponent je zvětšené písmo, které upraví ve vlastnosti této komponenty Font.

Neměl by být pro studenta problém vytvořit událost OnClick komponenty button. Nyní nastává důležitý okamžik, kdy student zjistí, že je třeba pracovat i s datovými typy. Jelikož

se zadávaná čísla vkládají do komponent Edit, jsou tato čísla datového typu string (řetězce) a nelze s nimi počítat. Student tedy bude muset obsahy těchto komponent převést na číslo. Převádění mezi datovými typy bylo popsáno ve studijním článku Práce s komponenty. Jakmile student ví, jak převádět datové typy, může se pustit do samotného sčítání čísel. Opět se tato situace dá řešit dvěma způsoby. První je, že lze celé počítání i vypisování výsledku zapsat do jednoho příkazu. To ale nebude nejvhodnější způsob pro začínajícího programátora. Student by si měl nejprve vytvořit lokální proměnné typu integer, do kterých poté bude ukládat hodnoty z textových polí, které bude rovnou převádět do typu integer, jelikož tyto hodnoty jsou typu string. Následně čísla sečte do třetí lokální proměnné a tuhle proměnou vypíše do třetího textového pole. Opět nesmí zapomenout na změnu datového typu.

V návrhu řešení student nalezne popis práce, kde je vše vysvětleno krok za krokem. Jelikož se jedná o tak jednoduchý případ, nejsou zde ukázky kódů, ale jen vypsány příkazy. Na konci návrhu řešení je k dispozici hotové řešení, které si student může stáhnout a otevřít.



Obrázek 16: Cvičení Součet dvou čísel – zadání

6.5 CVIČENÍ SOUČET DVOU ČÍSEL – ROZŠÍŘENÍ

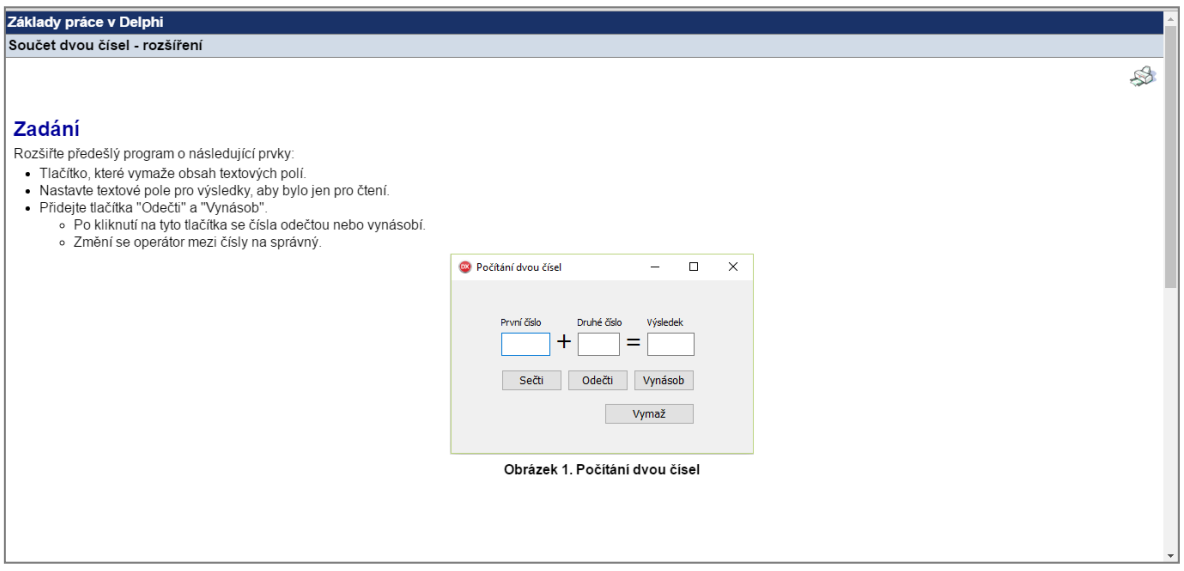
Cílem tohoto cvičení je rozšířit již nabyté zkušenosti s vlastnostmi komponent Edit, Label a Button. Zadáním tohoto cvičení, je rozšíření předešlého programu tak, že nebude umět jen sčítat, ale i odčítat a násobit. Bude tedy třeba upravit vzhled programu tak, že se přidají další dvě tlačítka do programu, a to právě pro odečítání a pro násobení.

V zadání se také požaduje, aby se při provádění výpočtu změnilo i znaménko mezi čísly podle daného výpočtu. Navíc program obsahuje tlačítko pro vymazání všech textových polí. Posledním požadavkem je, aby textové pole pro výsledky bylo jen pro čtení.

V typech řešení je pro studenta nápověda, jak se vypořádat s tím, aby textové pole bylo jen pro čtení. To není studentovi sdělen přímo, jelikož při hledání správné vlastnosti si zkušenosti jen rozšíří. Navíc taková to vlastnost je popsána i v tabulce nejběžnějších vlastností ve studijním článku o práci s komponenty.

Princip práce je naprosto stejný jako u předešlého cvičení jen s rozdílem, že student musí pracovat se čtyřmi tlačítky, to znamená, že i se čtyřmi událostmi OnClick komponent Button. Po vytvoření vzhledu bude student pracovat na událostech OnClick. Událost OnClick ve které se dvě čísla sčítají již má hotovou z minulého cvičení. Tu rozšíří o změnu znaménka mezi čísly. Jde o změnu vlastnosti Caption dané komponenty Label. Tento samý kód bude i v událostech, které budou čísla odčítat a násobit, jen s tím rozdílem, že se změní operátory na správné. Nakonec v události OnClick tlačítka „Vymaž“ bude změna vlastností Text komponent Edit na ‘ ‘.

V návrhu je popsána jak tvorba vzhledu, tak zdrojového kódu. Opět jde o jednoduchý příklad a nejsou zde ukázky zdrojových kódů, pouze jednotlivé příkazy. Na konci řešení je k dispozici i hotové řešení, které si student může stáhnout.



Obrázek 1. Počítání dvou čísel

Obrázek 17: Cvičení Součet dvou čísel – rozšíření

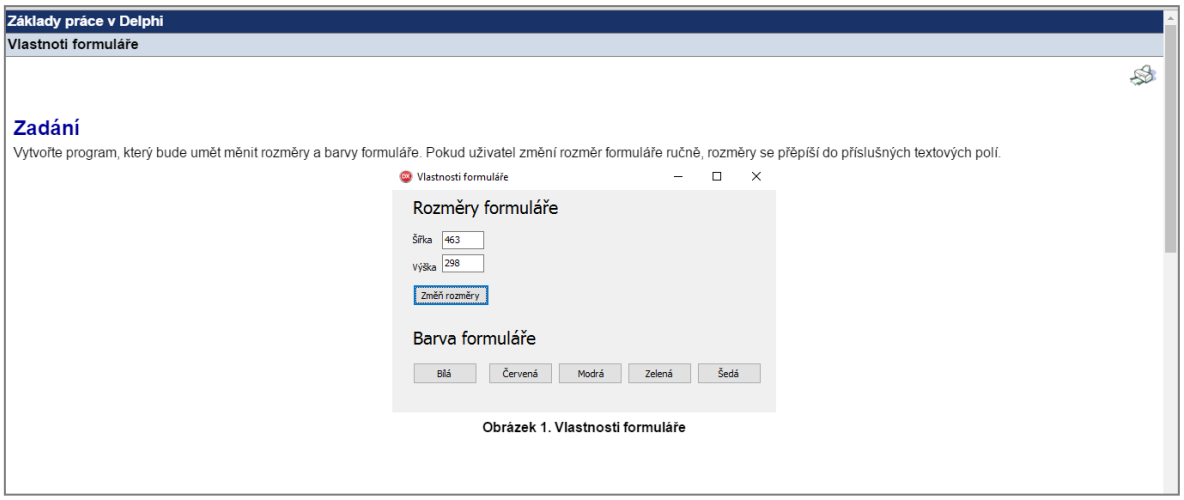
6.6 ÚKOL VLASTNOSTI FORMULÁŘE

Tímto úkolem by měl student prokázat, že rozumí práci s vlastnostmi a událostmi komponent a formuláře. To student prokáže tím, že zvládne pracovat s již známými komponenty Label, Edit a Button a s jejich vlastnostmi jako Text, Caption nebo Font nebo událostmi jako OnClick. Dále student prokáže znalost principů práce s vlastnostmi a událostmi, pokud zvládne měnit rozměr a barvy formuláře. S tímto se student dosud nesetkal, nic méně dostatek podkladů k vyřešení tohoto problému, které nalezne v typech pro řešení. K zadání je opět obrázek vzhledu programu a animace, která představuje funkci programu. Navíc se předpokládá, že v případě kdy si student nebude vědět rady, nahlídne zpět do potřebných studijních článků. Doporučený čas na vypracování tohoto úkolu je jedna hodina.

Jak již bylo výše popsáno, u úkolů nemají studenti k dispozici řešení, a měl by sloužit jako praktický test studenta. Úkolem studenta, je vytvořit program, který bude měnit vlastnosti formuláře. Konkrétně to jsou rozměry a barvy formuláře. Rozměry se mění za pomoci vlastnosti formuláře Width (šířka)a Height (výška) a barvy pomoci vlastnosti Color. To se bude provádět za pomoci komponent Edit, Button a Label.

U tohoto úkolu jsou vypsána i hodnotící kritéria. Mezi tato kritéria patří změna velikosti formuláře, změna barvy formuláře, práce s komponenty a funkčnost událostí. Kritérium změny velikosti formuláře je hodnoceno čtyřmi body, kdy se hodnotí změna šířky a výšky formuláře po dvou bodech. Kritérium změna barvy formuláře, je hodnoceno pěti body, a to po bodu za každou barvu. U kritéria práce s komponenty se myslí zvládnutí práce s komponenty Button, Edit a Label, a to po bodu za každý druh komponenty. Posledním kritériem je funkčnost událostí. Tím se myslí událost OnClick na tlačítka, která se hodnotí jedním bodem. Zbylé dva body připadají k nové událost formuláře ReSize, které student dostane při vyřešení zápisu velikosti formuláře do textových polí, pokud se změní velikost formuláře ručně.

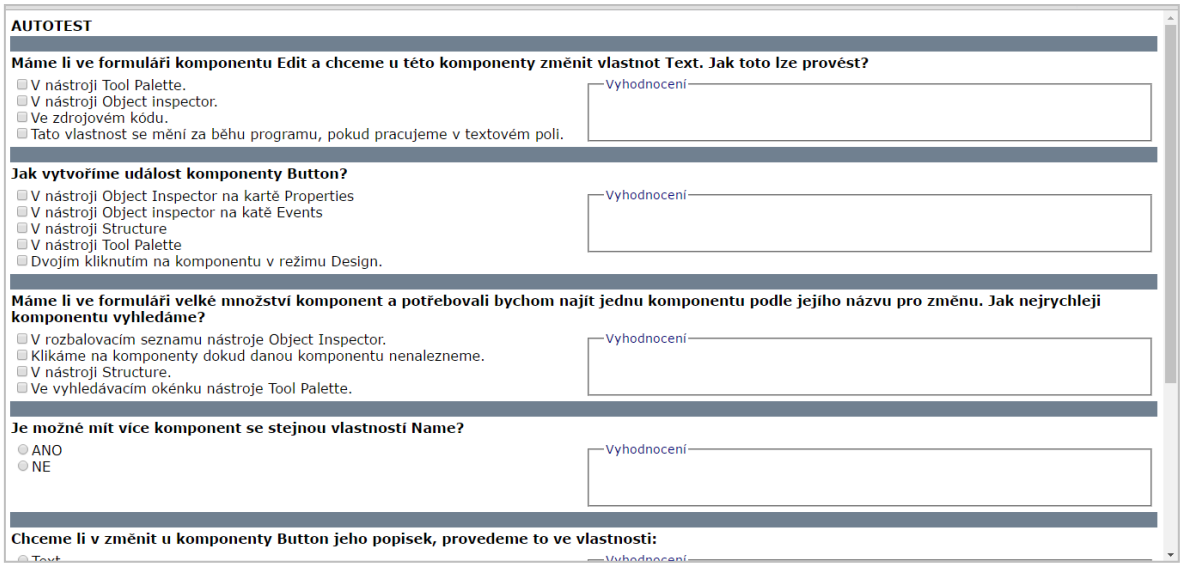
Pro Tutora je hotové řešení dispozici v řešení úkolu.



Obrázek 18: Úkol Vlastnosti formuláře

6.7 AUTOTEST ZÁKLADY PRÁCE V DELPHI

Tento autotest si klade za cíl, prověřit studentovu znalost základních principů práce ve vývojovém prostředí Delphi. Student díky autotestu dostane po odevzdání zpětnou vazbu o správnosti jeho odpovědí. V tomto autotestu je připraveno šest otázek, které jsou zaměřeny na práci s nejdůležitějšími nástroji vývojového prostředí Delphi, na práci s komponenty a s jejich vlastnostmi a událostmi. Tento test se skládá z otázek dichotomických (jedna správná odpověď ze dvou možných, například Ano-Ne), polytomických (jedna správná odpověď z více nabízených) a výčtových (umožňují více správných odpovědí). U některých otázek je poskytnuto i vysvětlení, pokud student odpoví na danou otázku chybně.



Obrázek 19: Autotest Základy práce v Delphi

6.8 KAPITOLA PRÁCE V DELPHI

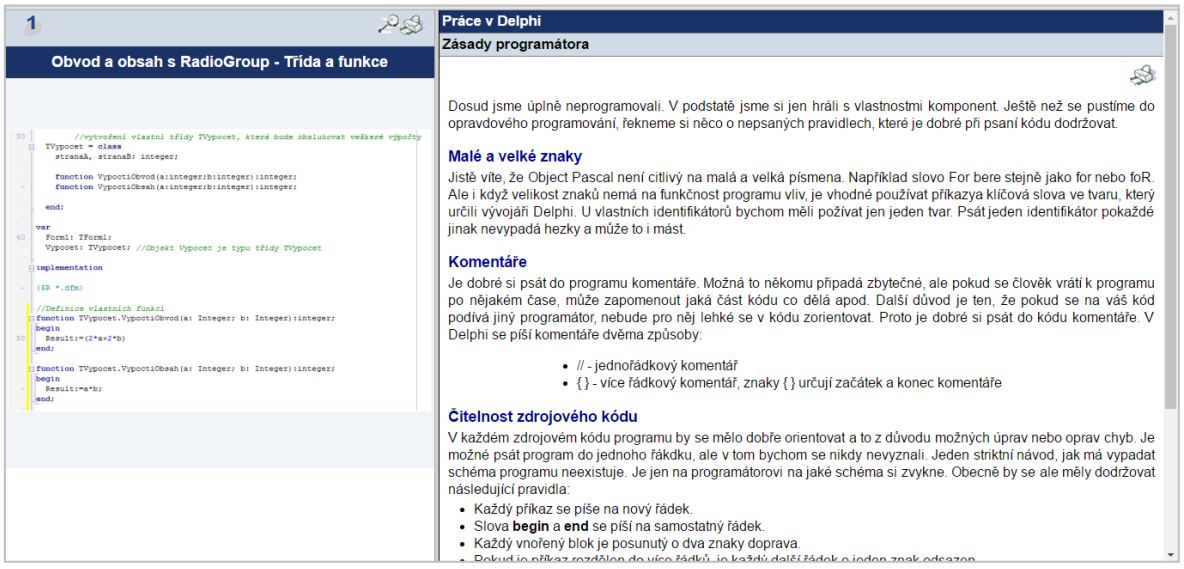
Tato kapitola se již nezabývá pouze prací s vlastnostmi a událostmi komponent. Cílem této kapitoly, je naučit studenty vytvářet programy za pomoci vlastních objektů, které se budou starat o běh programu. Dalším cílem je, že student bude vytvářet své programy dle zásad objektově orientovaného programování, kdy bude vhodně využívat rysy OOP. Dále se tato kapitola klade za cíl, že student bude svůj zdrojový kód vytvářet dle dobrých zásad programátora. Cílem této kapitoly, je také seznámit studenta s dalšími komponenty, které se student naučí využívat ve svých programech. Posledním neméně důležitým cílem, je rozšířit studentovi algoritmické myšlení, které využije při dalším programování. Ke konci této kapitoly se předpokládá, že student plně porozumí principům práce ve vývojovém prostředí Delphi a bude dále schopen se samostatně rozvíjet.

Do této kapitoly student vstupuje se znalostí objektově orientovaného programování, má zkušenosti s programováním konzolových programů z předmětu Programování 1 a dokáže pracovat s komponenty a jejich vlastnostmi a událostmi. Nyní student všechny tyto znalosti a zkušenosti využije v této kapitole. Student bude vytvářet ve svých programech třídy, procedury, funkce nebo konstruktory. Z těchto tříd bude vytvářet objekty, se kterými bude dále pracovat. Student se seznámí s novými komponenty jako je ListBox, ComboBox, RadioGroup, SpinEdit a s mnoha dalšími. Ke konci kurzu se student seznámí i s možností práce se soubory.

Student se v této kapitole setká s osmi cvičeními, která studenta rozvíjí studentovi schopnosti a dovednosti. Každé z cvičení je zaměřeno na danou probíranou oblast. Tato kapitola však není jen o cvičení i o výkladu nové problematiky. Proto student v této kapitole nalezne šest studijních článků, které se týkají problematiky, kterou si student lehce nedokáže vyzkoušet sám. Tato kapitola je obohacena o dva praktické úkoly, které prověřují studentovi dovednosti a znalosti. Opět je naprostou nezbytností, aby student zároveň s kurzem pracovat ve vývojovém prostředí Delphi, jelikož jedině praxí si student dokáže osvojit učivo této kapitoly.

6.9 STUDIJNÍ ČLÁNEK ZÁSADY PROGRAMÁTORA

Ještě než se v této kapitole začneme zabývat programováním jako takovým, je tu tento studijní článek, který pojednává o dobrých zásadách programátora. Cílem tohoto studijního článku, je předat studentovi rady, kterými by se měl držet při tvorbě svých zdrojových kódů, např. by měl dodržovat malá a velká písmena, ačkoli programovací jazyk Object Pascal nepozná rozdíl mezi velkými a malými písmeny. Jde pouze o přehlednost, např. pokud student deklaruje proměnnou, která má v názvu jen malá písmena, měl by to dodržovat i po zbytek práce s touto proměnnou. Dalším nepsaným pravidlem je, že názvy tříd začínají velkým písmenem T. Toto pravidlo by si měl student osvojit a dodržovat ho. Dalším dobrým pravidlem, je používat komentáře. Je pravdě podobné, že se student bude ke svým programům vracet, nebo bude své programy poskytovat někomu jinému. Bez komentářů bude těžké, jak pro programátora, tak pro někoho jiného, se v kódu zorientovat. Jednou z nejdůležitějších zásad, je však udržet čitelnost kódu, a to za pomoci vhodného řádkování a odsazení jednotlivých částí kódu. Z tohoto důvodu je tato problematika v tomto článku také popsána.



Obrázek 20: Studijní článek Zásady programátora

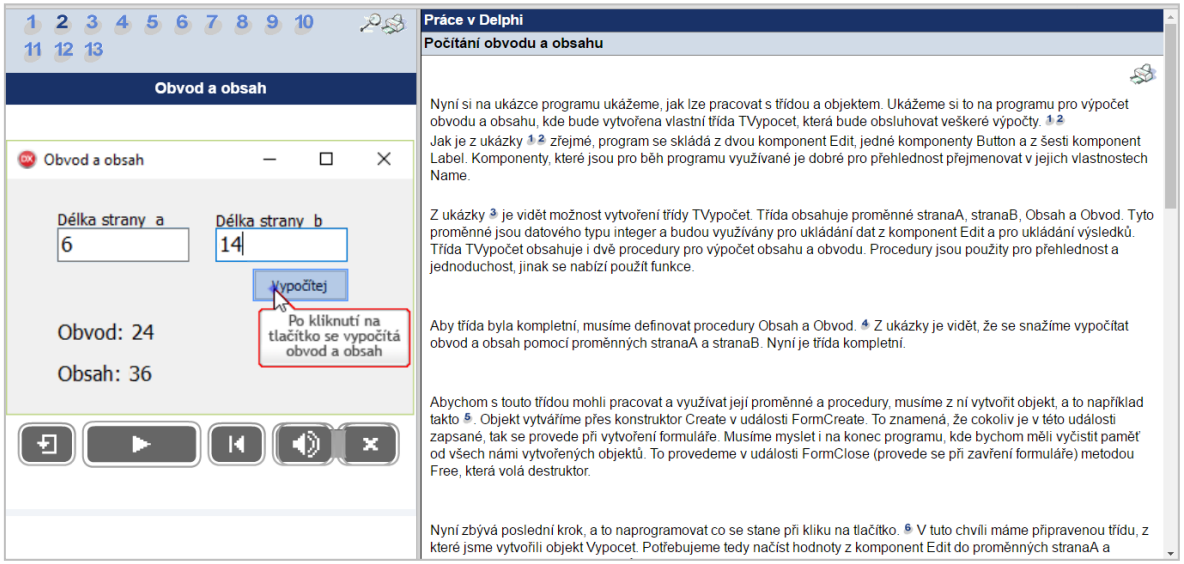
6.10 STUDIJNÍ ČLÁNEK POČÍTÁNÍ OBVODU A OBSAHU

Cílem tohoto studijního článku, je seznámit studenta s vytvářením programů, ve kterých pracujeme nejen s vlastnostmi komponent, ale také s vlastními objekty, které zajišťují hlavní činnost programu.

Tento studijní článek je řešen obdobně, jako studijní článek První program v předchozí kapitole. V tomto studijním článku se snažíme vytvořit jednoduchý program, který dokáže spočítat obvod a obsah čtyřúhelníku. Na tomto příkladě je ukázáno a krok po kroku popsáno, jak se tento program dá řešit. Předpokládá se, že studenti pracují zároveň ve vývojovém prostředí Delphi.

V demonstrační části článku, je obrázek vzhledu programu, podle kterého si studenti vytvoří vhled programu. Dále je k dispozici animace, představující funkci programu. Nejdůležitějším na tomto příkladu je, předvést studentům jak pracovat s vlastní třídou a objektem. Tato třída bude obsahovat dvě procedury, které obstarávají výpočet obvodu a obsahu. Procedury jsou zvoleny pro jednoduchost, jinak by se nejlépe hodily funkce s parametry. Ukázku deklarace třídy, mají studenti k dispozici v demonstrační části článku. Druhým krokem, je definovat činnost procedur, pro výpočet obvodu a obsahu. Opět je ukázka definice procedur k dispozici v demonstrační části programu. V této fázi mají studenti připravenou třídu, ze které si deklarují a vytvoří objekt. Tento objekt vytvoří v události formuláře OnCreate. Tímto se studenti seznamují s událostí formuláře OnCreate, která nastává při jeho vytvoření, v našem případě i při vytvoření programu. Tento objekt je vytvořen defaultním konstruktorem Create. Následně je objekt zrušen v události formuláře OnClose. Tímto se také studenti seznamují s touto událostí formuláře, která nastane při zavření formuláře (při zavření programu). Studenti opět naleznou ukázku vytvoření a zrušení objektu v demonstrační části článku. Nyní je vše připraveno pro výpočet. Poslední krok se bude odehrávat v události OnClick komponenty Button. V této události se přiřadí hodnoty příslušných vlastností textových polí do proměnných. Dále se volají procedury pro výpočet. Nakonec je třeba vypočtené hodnoty zaspat do příslušných komponent. Ukázka této události je také k dispozici v demonstrační části článku. Ve výkladové části článku je možnost stažení hotového řešení.

Tímto programem, si student vyzkoušel jak pracovat s vlastnostmi komponent společně s objekty.



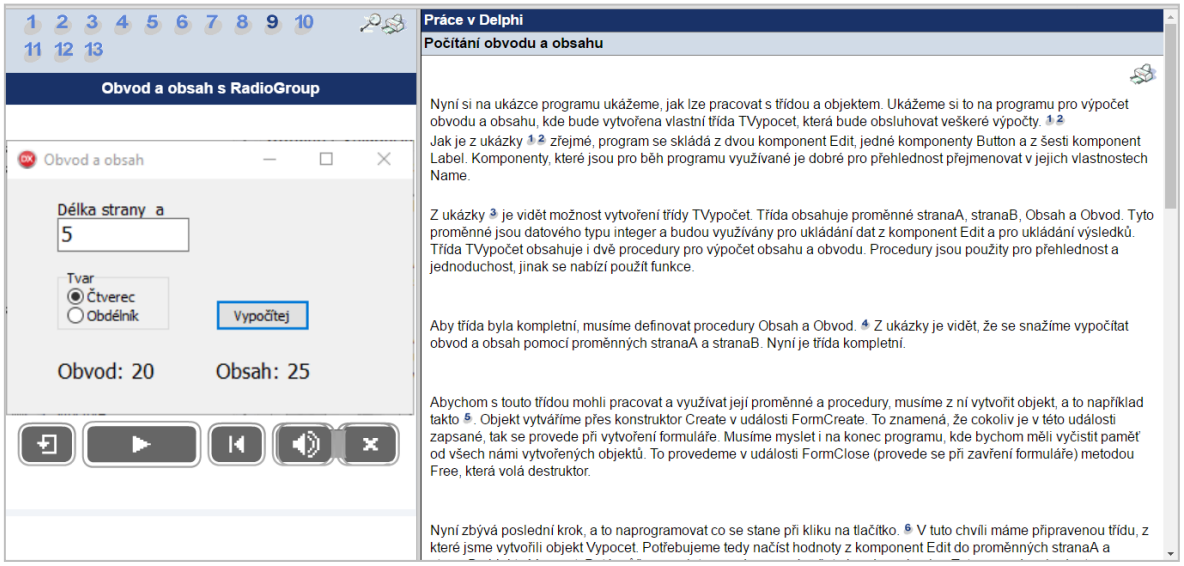
Obrázek 21: Studijní článek Počítání obvodu a obsahu

Tímto programem tato kapitola nekončí. Následuje druhý program, který rozšiřuje program první. Tento program bude počítat také obvod a obsah, jen bude rozlišovat, zda se bude počítat obvod a obsah čtverce nebo obdélníku. Cílem tohoto programu, je seznámit studenty s novou komponentou RadioGroup, která představuje skupinu radiobuttonů. Student se seznámí s hlavními vlastnostmi této komponenty. K tomuto účelu slouží animace v demonstrační části článku, která představuje práci s položkami této komponenty. Druhým cílem je, aby student v tomto programu využil funkcí s parametry místo procedur. Tím student pozná rozdíl v efektivnosti použití procedury a funkce s parametry. Nakonec se v tomto programu student seznámí s vlastností Visible. Opět je vytvoření tohoto programu popsáno ve studijním článku krok za krokem, stejně jak tomu bylo u programu prvního.

Student má opět k tomuto úkolu ukázku vzhledu, podle které vytvoří vzhled svého programu. Následuje animace, ve které je předvedena funkce programu. Funkce spočívá v tom, že pokud máme vybráno, že chceme počítat obvod a obsah čtverce, je viditelné jen jedeno textové pole pro zadání hodnoty strany a. Je tomu tak, jelikož pro výpočet hodnot čtverce více nepotřebujeme. Jakmile v programu zvolíme, že chceme počítat hodnoty obdélníka, zobrazí se nám druhé textové pro zadání hodnoty strany b. Volbu, zda chceme počítat hodnoty čtverce nebo obdélníka zajišťuje komponenta RadioButton.

Je důležité, aby si uvědomil, že je třeba vytvořit kompletní vzhled programu a poté zavedl defaultní vzhled, to znamená nastavit v komponentě RadioGroup označení první položky „Čtverec“ a skrýt druhé textové pole. To provede změnou vlastnosti Visible na „False“. Defaultní vzhled student nastaví buď v nástroji Object Inspector, nebo v kódu v události při vytvoření formuláře. Po vytvoření vzhledu student obstará zobrazení a skrývání druhého pole při změně položky komponenty RadioGroup. To student provede v události OnClick komponenty RadioGroup. V této události bude zjišťovat, jaká položka je označena, pokud je označena položka „Obdélník“, zobrazí se druhé textové pole, a to přiřazením hodnoty „True“ do vlastnosti Visible komponenty Edit2.

Po vytvoření vzhledu je třeba upravit třídu z minulého programu, konkrétně změníme procedury na funkce s parametry, do kterých se budou zapisovat hodnoty stran počítaného tvaru. Tyto procedury se musí změnit na funkce s parametry i v části kódu, kde se definuje jejich činnost. Posledním krokem v tomto programu, je změnit událost OnClick tlačítka „Vypočítej“. V této události student zjišťuje, jaká položka z komponenty RadioGroup je označena. Podle označené položky se volá příslušná funkce, do jejíchž parametrů se vkládá hodnota strany nebo stran. Výsledek funkce se zapíše do komponenty pro vypisování výsledků.

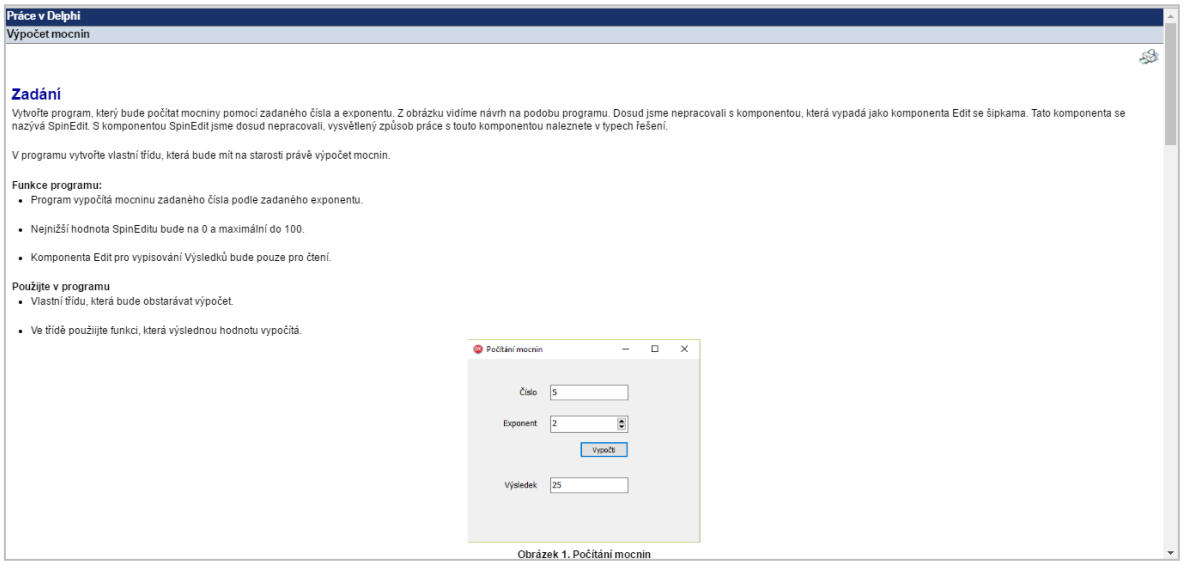


Obrázek 22: Studijní článek Počítání Obvodu a Obsahu - rozšíření programu

6.11 CVIČENÍ VÝPOČET MOCNIN

Prvním cvičením v této kapitole se jmenuje Výpočet mocnin. V tomto cvičení má student za úkol vytvořit program, který vypočítá výsledek podle zadaného čísla a exponentu. Cílem tohoto cvičení je, že si student utvrdí dovednosti a znalosti nabyté z minulého studijního článku. Druhým cílem je seznámit studenta s novou komponentou SpinEdit, která představuje číselný vstup do programu.

V zadání je po studentovi vyžadováno, že v programu bude využito vlastní třídy, která bude obstarávat výpočet. Dále se požaduje využití funkce pro samotný výpočet výsledné hodnoty. V zadání je taky požadavek, na omezení hodnot v komponentě SpinEdit, a že tato komponenta bude pouze pro čtení. Pod tímto zadáním student nalezne ukázku vzhledu programu a animaci, prezentující funkci programu. Nakonec pod ukázkou vzhledu a funkce student nalezne typy pro řešení, ve kterých je popsána nová komponenta SpinEdit, jeho nejdůležitější vlastnosti a typ jak se dopočítat výsledku.



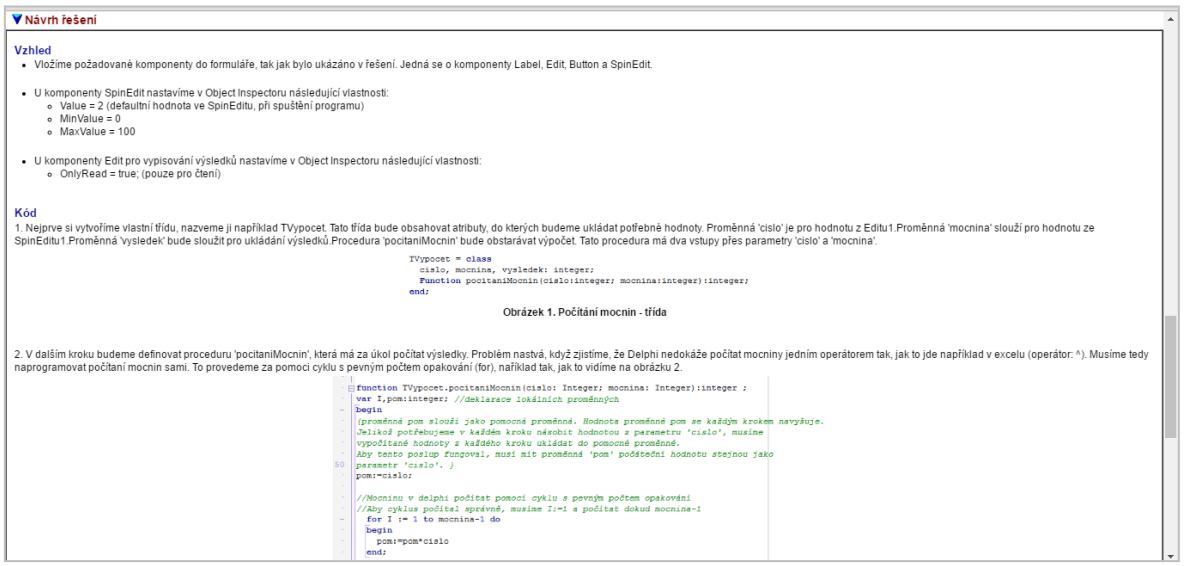
Obrázek 23: Cvičení Počítání mocnin – zadání

Student jako první vytvoří vzhled podle ukázky ze zadání. Seznámí se nově s komponentou SpinEdit a nastaví její vlastnosti MinValue a MaxValue, a to pro minimální a maximální povolenou hodnotu v této komponentě tak, jak to bylo vyžadováno v zadání. Zbytek použitých komponent již student zná a zvládá s nimi pracovat.

Nyní začne student pracovat na zdrojovém kódu. Jako první věc, kterou musí v kódu udělat, je vytvořit vlastní třídu ve které budou potřebné atributy a také funkce, nejlépe s parametry pro číslo a exponent. Aby student dokončit vlastní třídu, je třeba, aby

se pustil do stěžejní části kódu, a to vytvořit funkci pro výpočet výsledné hodnoty. Student zjistí, že v Object Pascalu neexistuje žádná funkce, která by dokázala vypočítat mocninu z čísla a exponentu. Je tedy na studentovi, aby samotný výpočet zařídil právě ve své funkci. Pro co nejvyšší univerzálnost bude právě dobré, pokud studentova funkce bude obsahovat i parametry pro číslo a exponent. Z principu výpočtu mocniny by měl student přijít na to, že je třeba cyklu s pevným počtem opakování. Tento cyklus se bude opakovat podle zadaného exponentu. V těle cyklu se bude násobit zadané číslo. Po skončení cyklu se výsledek vypíše do výstupu funkce. Poté má student třídu vytvořenou a je třeba vytvořit z ní objekt. To provede nejlépe v události formuláře OnClick a následně zařídí zrušení objektu, a to nejlépe při zavření formuláře neboli v události formuláře OnClose. Posledním krokem při tvorbě tohoto programu, bude vytvoření události OnClick komponenty Button. V této události bude student vkládat hodnoty z komponent Edit a SpinEdit do vlastní funkce pro výpočet a následně výsledek funkce vypisovat do příslušeného textového pole (Edit2).

Pokud si student nebude jistý svým postupem, nebo si nebude vědět rady, je tu pro něj připraven návrh řešení. V tomto návrhu řešení je pro studenta připraven popis vytvoření programu. Student tedy projde krok za krokem části programu. Tyto části programu jsou doplněny o obrazové ukázky zdrojového kódu, které se k probíranému kroku vztahují. Na konci návrhu řešení je k dispozici i hotové řešení, které si student může stáhnout a spustit si hotový projekt ve svém počítači.



Obrázek 24: Cvičení Počítání mocnin - návrh řešení

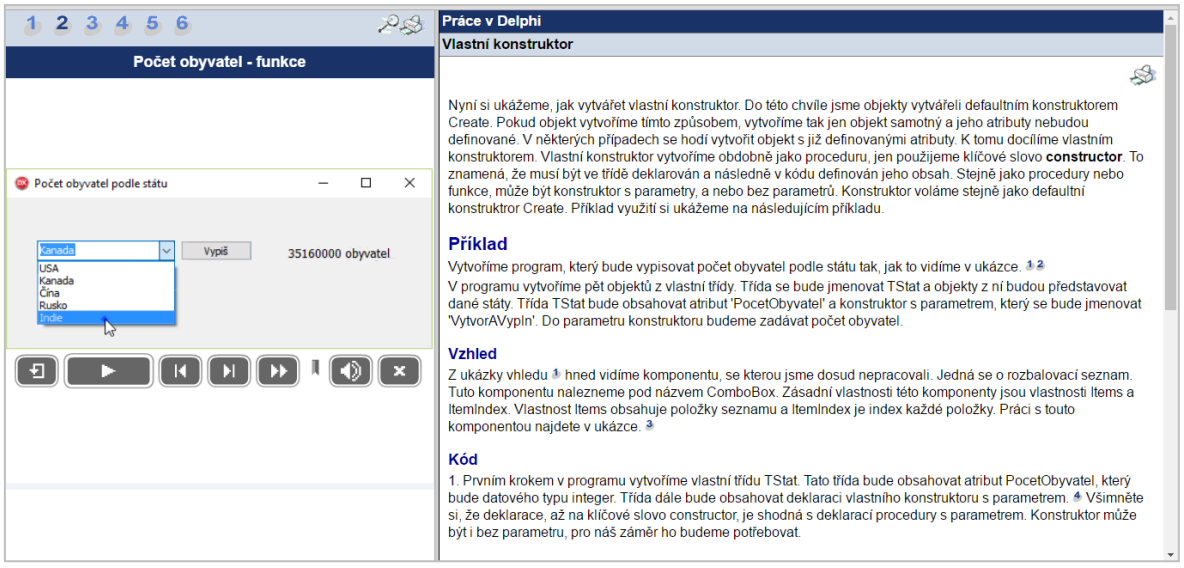
6.12 STUDIJNÍ ČLÁNEK VLASTNÍ KONSTRUKTOR

Jak je z názvu patrné, tento studijní článek se zabývá prací s vlastními konstruktory. Cílem tohoto článku, je seznámit studenty s tvorbou objektu za pomoci vlastních konstruktorů. Student by tímto článkem měl pochopit význam vlastních konstruktorů a v další práci vhodně volit mezi defaultním konstruktorem Create a vlastním konstruktorem. Student by si tímto článkem měl uvědomit, že při použití vlastního konstruktoru můžeme pracovat s objektem ihned při jeho vytvoření. Např. chceme-li definovat atributy objektu, nemusíme je po vytvoření definovat v nějaké události nebo v nějaké proceduře, to vše může provést vlastní konstruktor, kterým objekt vytvoříme. Druhým hlavním cílem, je seznámit studenta s novou komponentou, a to komponentou ComboBox, která představuje rozbalovací seznam. Opět se předpokládá, že student bude pracovat zároveň se studijním článkem ve vývojovém prostředí Delphi.

Vytvoření vlastního konstruktoru, je představeno opět na příkladu, a to na programu, který bude vypisovat počet obyvatel pěti vybraných států. Každý stát bude představovat objekt, který bude mít jeden jediný atribut, a to počet obyvatel. V demonstrační části článku, je představen vzhled a pomocí aplikace i funkce programu. Na první pohled student vidí, že v programu je nová komponenta, s kterou dosud nepracoval. Je to komponenta ComboBox neboli rozbalovací seznam. Z této komponenty bude student vybírat stát, ze kterého chce vypsat počet obyvatel. Práce s komponentou ComboBox je naprosto stejná jako práce s RadioGroup, se kterou se student již setkal a pracoval s ní.

První činnost studenta, je vytvoření vzhledu. Vzhled tohoto programu je jednoduchý, skládá se jen z komponenty Label, Button a nové komponenty ComboBox. Práce s novou komponentou ComboBox, je představena jak ve výkladu, tak v demonstrační části, za pomoci animace. Po vytvoření vzhledu a nastavení komponenty ComboBox, může student přejít k tvorbě zdrojového kódu. Student ví, že bude vytvářet pět stejných objektů. Neměl by tedy uvažovat o tvorbě pěti stejných tříd, ale jen o jedné třídě, ze které vytvoří pět objektů. V této třídě bude deklarován atribut pro počet obyvatel a vlastní konstruktor s parametrem. Do tohoto parametru bude student při volání vkládat počet obyvatel. Tato hodnota se v konstruktoru přiřadí do atributu objektu. Následně v události formuláře OnCreate budeme vytvářet pět objektů, představující vybrané státy. Student

bude tyto objekty vytvářet pomocí vlastního konstruktoru. Poté vytvoří událost OnClick komponenty Button. V této události bude student vypisovat počet obyvatel objektů. To, z jakého objektu se bude atribut vypisovat, bude rozhodnuto podmínkami, které budou zjišťovat, jaký stát byl v rozbalovacím seznamu vybrán. Toto rozhodování by se dalo řešit i pomocí Casu. Ke konci programu student vytvoří i událost formuláře OnClose, ve které vytvořené objekty zruší. Tyto všechny kroky jsou v tomto článku popsány a vysvětleny. Nakonec má student i k dispozici hotové řešení, které si může stáhnout a spustit ve svém vývojovém prostředí.



Obrázek 25: Studijní článek Vlastní konstruktor

6.13 STUDIJNÍ ČLÁNEK LISTBOX

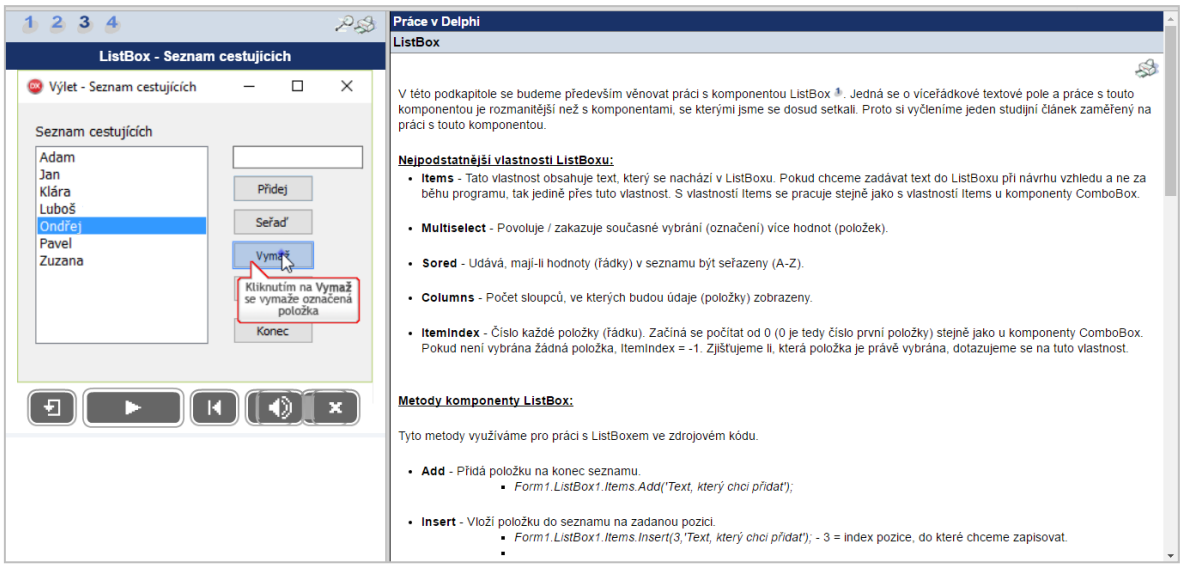
Tento studijní článek se zabývá pouze komponentou ListBox, která představuje více řádkové textové pole. Je tomu tak, protože práce s touto komponentou je o něco rozsáhlejší, než s komponenty, se kterými se student dosud setkal. Cílem tohoto studijního článku, je seznámit studenta s komponentou ListBox. Student se také naučí používat základní vlastnosti a základní příkazy pro práci s touto komponentou.

V tomto článku se student dozví, co komponenta ListBox představuje, a co představují jeho nejhlavnější vlastnosti. Zde si student uvědomí, že jde již o třetí komponentu, která svůj obsah ukládá v řádcích ve vlastnosti Items a konkrétní položka se volí vlastností ItemIndex. Práce s těmito vlastnostmi je znázorněna v animaci, kterou student nalezne opět v demonstrační části studijního článku. S těmito vlastnostmi se student setkal již při práci s komponenty RadioGroup a ComboBox. Tyto komponenty mají společné i další

některé vlastnosti a dokonce i příkazy, které umožňují pracovat s položkami komponenty přímo v kódu. Nicméně je přeci komponenta ListBox od těchto dvou odlišná, především svým významem. Dále se student seznámí s příkazy, pomocí kterých bude student s obsahem této komponenty pracovat přímo v kódu.

Pro vyzkoušení práce s touto komponentou je připraven příklad. Opět se očekává studentovo zapojení v podobě práce ve vývojovém prostředí. Student bude vytvářet společně s výkladem program, který bude umět do komponenty ListBox zapisovat účastníky zájezdu a následně tento seznam cestujících editovat. Konkrétně tento program bude umět zapisovat položky, řadit položky od A-Z, mazat položky nebo vymazat celý seznam. Nakonec bude v programu tlačítko „Konec“, kterým program zavře. Vzhled a funkce programu, je znázorněna na obrázku a animaci v demonstrační části článku. Cílem tohoto programu je, aby si student prakticky vyzkoušel pracovat s co nejvíce příkazy spojených s komponentou ListBox.

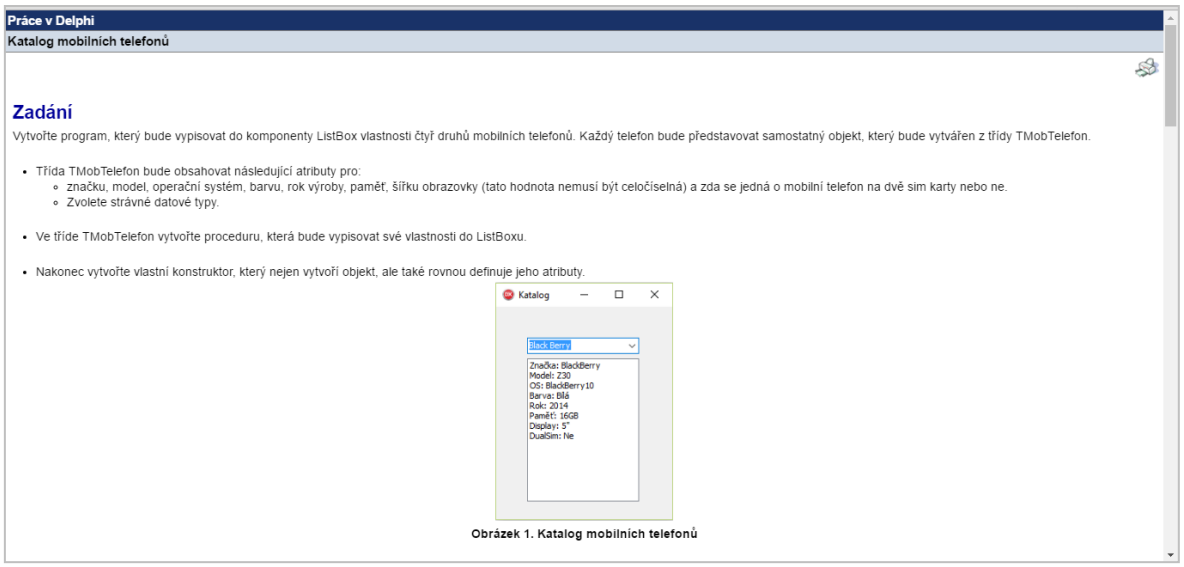
Práce studenta na tomto programu je následující. Jako první student vytvoří vzhled, který se skládá z komponent Edit, Button, Label a hlavně z nové komponenty ListBox. Poté bude pracovat pouze v událostech OnClick tlačítek, které do formuláře vložil. V těchto událostech bude student používat vhodné příkazy pro práci s položkami komponenty ListBox. Student si tedy vyzkouší příkazy Add, Sorted, DeleteSelected a Clear. Pro ukončení programu využije příkazu Close. Všechny tyto kroky, jsou popsány a vysvětleny ve výkladové části článku. Jako opora pro tuto část výkladu, slouží ukázka zdrojového kódu, která je ke zhlednutí v demonstrační části. Na konec je k dispozici i hotové řešení, které lze stáhnout a spustit v Delphi.



Obrázek 26: Studijní článek ListBox

6.14 CVIČENÍ KATALOG MOBILNÍCH TELEFONŮ

Cílem tohoto cvičení, je utvrdit studentovy znalosti a dovednosti v oblasti vytváření objektů pomocí vlastních konstruktorů. Dále je cílem, aby si student procvičil práci s komponentou ListBox a ComboBox. Posledním cílem je seznámit studenta s událostí OnChange.



Obrázek 27: Cvičení Katalog mobilních telefonů – zadání

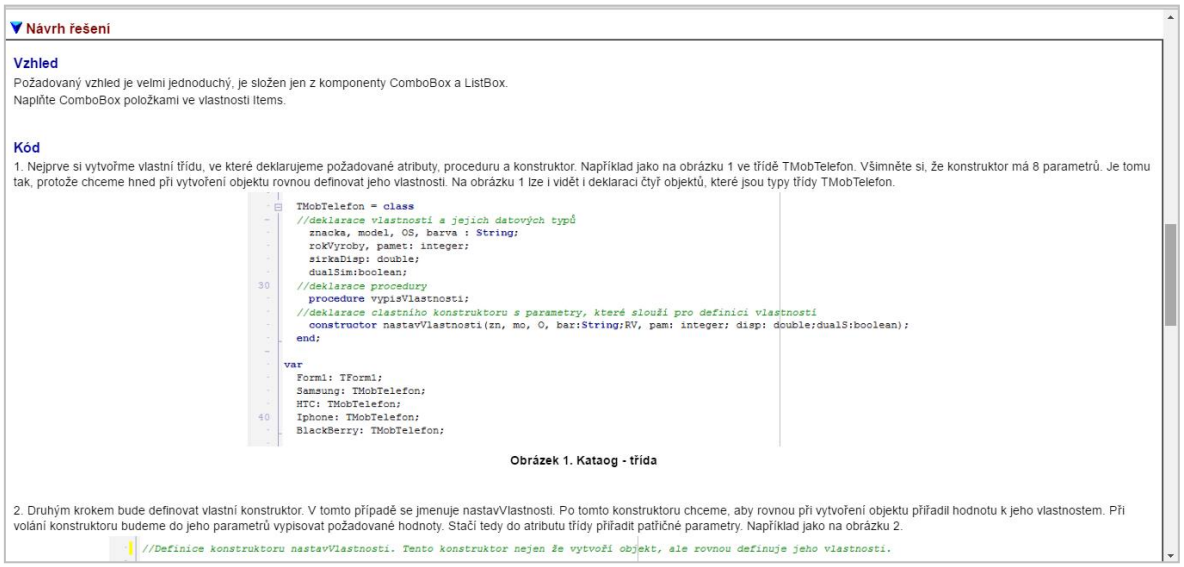
V tomto cvičení, má student za úkol vytvořit program, který bude do komponenty ListBox vypisovat vlastnosti mobilních telefonů. Druh mobilního telefonu si uživatel vybere pomocí rozbalovacího seznamu. Vlastnosti se do komponenty ListBox vypíší hned po výběru položky z rozbalovacího seznamu. Jedná se tedy o obdobné využití vlastního

konstruktoru, jako ve studijním článku Vlastní konstruktor. V zadání je po studentovi požadováno, aby ve svém programu využil vlastní třídu TMobTelefon, ve které budou atributy pro vlastnosti mobilního telefonu. Požadované atributy jsou vypsány. Dále tato třída bude obsahovat proceduru pro výpis vlastností do komponenty ListBox a vlastní konstruktor, který bude mít tolik parametru, jako má třída atributů. Tento konstruktor bude při vytváření objektu rovnou definovat jeho atributy. Student má pod zadáním opět obrazovou ukázku vzhledu programu a animaci, představující funkci programu. V typech řešení, je snaha nasměrovat studenta správným směrem. Jelikož chceme tímto kurzem vést studenta k pochopení principů práce v Delphi a chceme ho přimět ke zkoušení nových nepoznaných věcí, dáváme zde prostor pro to, aby student sám přišel na událost OnChange komponenty ComboBox. Pokud na tuto událost nepřijde, měl by se s touto situací vypořádat nějakou jinou cestou.

Pro kontrolu své práce nebo pro pomoc při řešení, může student využít poslední část tohoto cvičení, a to návrh řešení. V tomto návrhu řešení najde student popis možného řešení, a to krok po kroku. Tyto kroky jsou doplněny ukázkami částmi kódu, které se k danému kroku vztahují.

Práce studenta na tomto cvičení začíná vzhledem programu. Jediné nastavování v nástroji Object inspector, které při vzhledu musí udělat, je nastavit položky v komponentě ComboBox. To by se dalo případně řešit i v kódu. Po vytvoření vzhledu si student vytvoří vlastní třídu, ve které budou deklarovány požadované atributy, procedura a konstruktor. Po vytvoření třídy vytvoří vlastní konstruktor s parametry. Tento konstruktor nebude dělat nic jiného, než přiřazovat hodnoty z parametrů do příslušných atributů. Dále je za potřebí, aby student vytvořil proceduru, která bude vypisovat své hodnoty do komponenty ListBox. Tato procedura má své výhody i nevýhody. Nevýhoda je taková, že je závislá na komponentě ListBox, která je deklarovaná v jiné třídě. Výhoda je taková, že jednou procedurou si ušetříme velké množství řádků při vypisování hodnot z objektů. V této chvíli student může vytvářet objekty za pomoci vlastního konstruktoru. To udělá nejlépe v události formuláře OnCreate neboli při vytvoření programu. Do parametrů konstruktoru vždy zadá hodnoty, které daný objekt má obsahovat. Nyní může student vlastnosti objektů vypisovat, a to při výběru položky v komponentě ComboBox. Jakmile vybereme jinou položku, než která byla vybrána, generuje se událost OnChange. V této

události se bude rozhodovat, z jakého objektu se hodnoty vypíší. To bude určovat právě vybraná položka v rozbalovacím seznamu. To jaká položka je vybrána, bude student zjišťovat buď podmínkami, nebo pomocí Casu. Nakonec student zařídí, aby se při zavření programu odstranily objekty z paměti.



Návrh řešení

Vzhled

Požadovaný vzhled je velmi jednoduchý, je složen jen z komponenty ComboBox a ListBox.
Naplňte ComboBox položkami ve vlastnosti Items.

Kód

1. Nejprve si vytvořme vlastní třídu, ve které deklarujeme požadované atributy, proceduru a konstruktor. Například jako na obrázku 1 ve třídě TMobTelefon. Všimněte si, že konstruktor má 8 parametrů. Je tomu tak, protože chceme hned při vytvoření objektu rovnou definovat jeho vlastnosti. Na obrázku 1 lze i vidět i deklaraci čtyř objektů, které jsou typy třídy TMobTelefon.

```
TMobTelefon = class
//deklarace vlastností a jejich datových typů
  znacka, model, OS, barva : String;
  rokVyroby, pamet: integer;
  sirkaDisp: double;
  dualSim:boolean;
//deklarace procedury
  procedure vypisVlastnosti;
//deklarace vlastního konstruktoru s parametry, které slouží pro definici vlastností
  constructor nastavVlastnosti(zn, mo, O, bar:String;RV, pam: integer; disp: double;dualS:boolean);
end;

var
  Form1: TForm1;
  Samsung: TMobTelefon;
  HTC: TMobTelefon;
  Iphone: TMobTelefon;
  BlackBerry: TMobTelefon;
```

Obrázek 1. Kataog - třída

2. Druhým krokem bude definovat vlastní konstruktor. V tomto případě se jmenuje nastavVlastnosti. Po tomto konstruktoru chceme, aby rovnou při vytvoření objektu přiřadil hodnotu k jeho vlastnostem. Při volání konstruktoru budeme do jeho parametrů vypisovat požadované hodnoty. Stačí tedy do atributu třídy přiřadit patřičné parametry. Například jako na obrázku 2.

```
//Definice konstruktoru nastavVlastnosti. Tento konstruktor nejen že vytvoří objekt, ale rovnou definuje jeho vlastnosti.
```

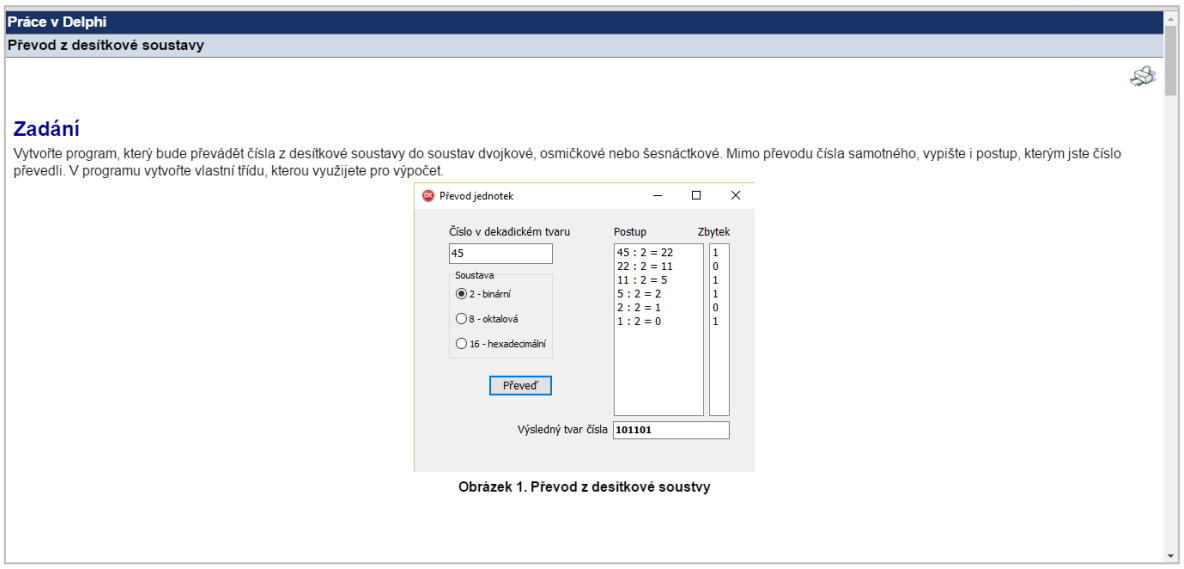
Obrázek 28: Cvičení Katalog mobilních telefonů - návrh řešení

6.15 CVIČENÍ PŘEVOD Z DESÍTKOVÉ SOUSTAVY

Studenti v tomto cvičení mají za úkol vytvořit program, který bude převádět číslo z desítkové soustavy do soustavy dvojkové, osmičkové a šestnáctkové. Zároveň se bude vypisovat postup převodu, a to i se zbytky, ze kterých se výsledné číslo složí.

Cílem tohoto cvičení, je utvrzení znalostí a schopností práce s již známými komponenty a s jejich vlastnostmi a událostmi. Konkrétně se jedná o komponenty Label, Edit, Button, RadioGroup a ListBox. Druhým cílem, je procvičení studentova logického myšlením. Posledním cílem, je připomenout studentovi princip převodu čísel z desítkové soustavy do jiných soustav.

Zadání cvičení zní následovně: „Vytvořte program, který bude převádět čísla z desítkové soustavy do soustav dvojkové, osmičkové nebo šestnáctkové. Mimo převodu čísla samotného, vypište i postup, kterým jste číslo převedli. V programu vytvořte vlastní třídu, kterou využijete pro výpočet“. K zadání má student k dispozici obrázek, představující vzhled aplikace a animaci, která představuje funkci programu. Nakonec zadání má student k dispozici typy pro řešení. Zde se student nedozví jak program řešit, ale nalezne zde jen rady ohledně převodu soustav.



Obrázek 1. Převod z desítkové soustvy

Obrázek 29: Cvičení Převod z desítkové soustavy – zadání

Prvním studentovým krokem při tvorbě programu, je vytvořit vzhled. Zde použije již zmíněné komponenty. Jediné nastavování v nástroji Object Inspector studenta čeká u komponent Edit a RadioGroup, kdy upraví obsahy těchto komponent.

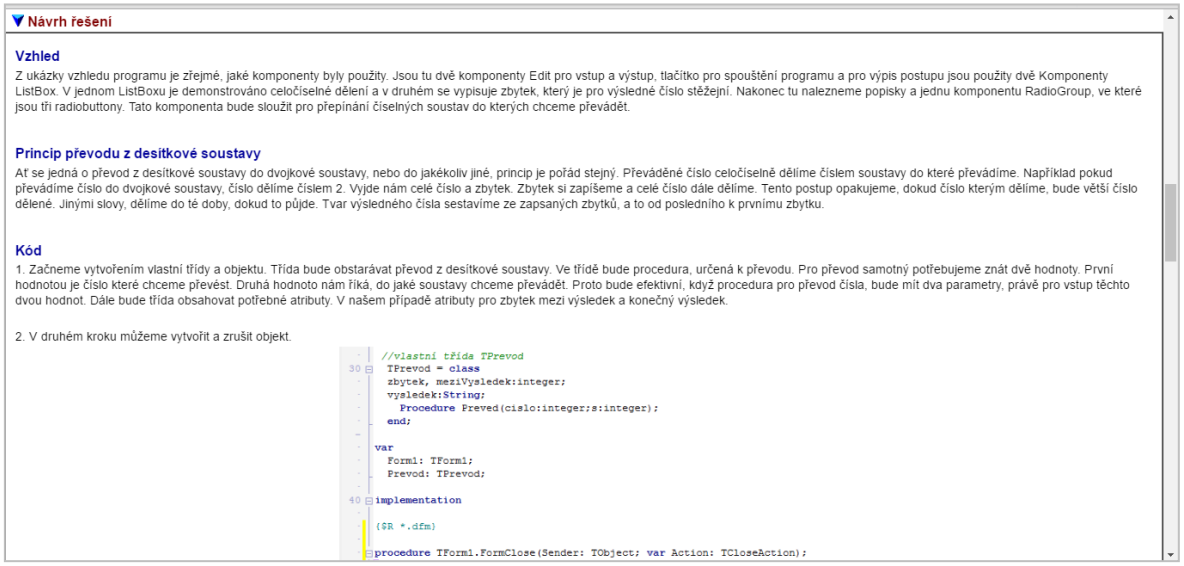
Po vytvoření vzhledu si student vytvoří vlastní třídu, která bude obstarávat převody soustav. Měl by student sám přijít na to, že bude dobré, když ve třídě budou potřebné atributy a procedura, která bude dané číslo převádět. Bylo by také dobré, kdyby tato procedura měla parametry pro převáděné číslo a číslo cílové soustavy. V tuto chvíli je již samozřejmostí, že si student bez váhání dokáže vytvořit a zrušit objekt ve vhodných událostech.

Nejtěžší a nejdůležitější částí tohoto programu, je procedura pro převod čísla. Tato procedura má za úkol číslo převést a vypsat postup i se zbytkem. V Object Pascalu existují funkce, které by číslo převedly do dané soustavy, ty ale nevypíší svůj postup a tak je nelze využít pro řešení této procedury. Student by měl znát princip převodu z desítkové soustavy do jiných soustav, jelikož bez této znalosti nebude schopen vytvořit tuto proceduru. Z principu převodu víme, že ať se jedná o převod z desítkové soustavy do dvojkové soustavy, nebo do jakékoliv jiné, princip je pořád stejný. Převáděné číslo celočíselně dělíme číslem soustavy, do které převádíme. Například pokud převádíme číslo do dvojkové soustavy, číslo dělíme číslem 2. Vyjde nám celé číslo a zbytek. Zbytek si zapíšeme a celé číslo dále dělíme. Tento postup opakujeme dokud číslo, kterým dělíme, bude větší než číslo dělené. Jinými slovy, dělíme do té doby, dokud to půjde. Tvar

výsledného čísla sestavíme ze zapsaných zbytků, a to od posledního k prvnímu zbytku. Pro tento účel se hodí cyklus a podmínkou na začátku. Jelikož se jedná o proceduru, která má převést a zapisovat, bude dobré, když si řešení této procedury rozdělí na dvě části, a to na převod a na zápis do komponent ListBox. Student v tomto cyklu zařídí celočíselné dělení čísla a napisování zbytku. Ze zbytku v každém kroku tvoříme i výsledek. Pro dobrou manipulaci s jednotlivými čísly ve výsledku, je dobré zbytek převést do datového typu string. Poté student zařídí vypisování do dvou komponent ListBox (první pro postup, druhý pro zbytek). Tyto dva příkazy je dobré umístit hned za vypočítání zbytku a mezivýsledku. Posledním krokem tohoto cyklu, je zjištění, zda není zbytek vyšší než číslo 9. Kdyby tomu tak bylo, jednalo by se o hexadecimální soustavu a zbytek vyšší než číslo 9 se zapisuje písmeny od A – F. Toto student bude zjišťovat před zápisem do výsledku. Poté se do výsledku zapíše písmeno místo čísla. Po skončení cyklu se výsledek vypíše do příslušného textového pole.

Posledním krokem, je událost OnClick tlačítka, kterým se spouští převod. Zde student bude zjišťovat, do jaké soustavy chceme převádět (podle označené položky v komponentě RadioGroup) a podle tohle toho bude volat proceduru pro převod se správnými parametry.

Pokud si student nebude vědět rady, může se obrátit na návrh řešení. V tomto návrhu je možný postup popsán a vysvětlen krok za krokem, společně s ukázkami částí kódu, které se k danému kroku vztahují.



Obrázek 30: Cvičení Převod z desítkové soustavy - návrh řešení

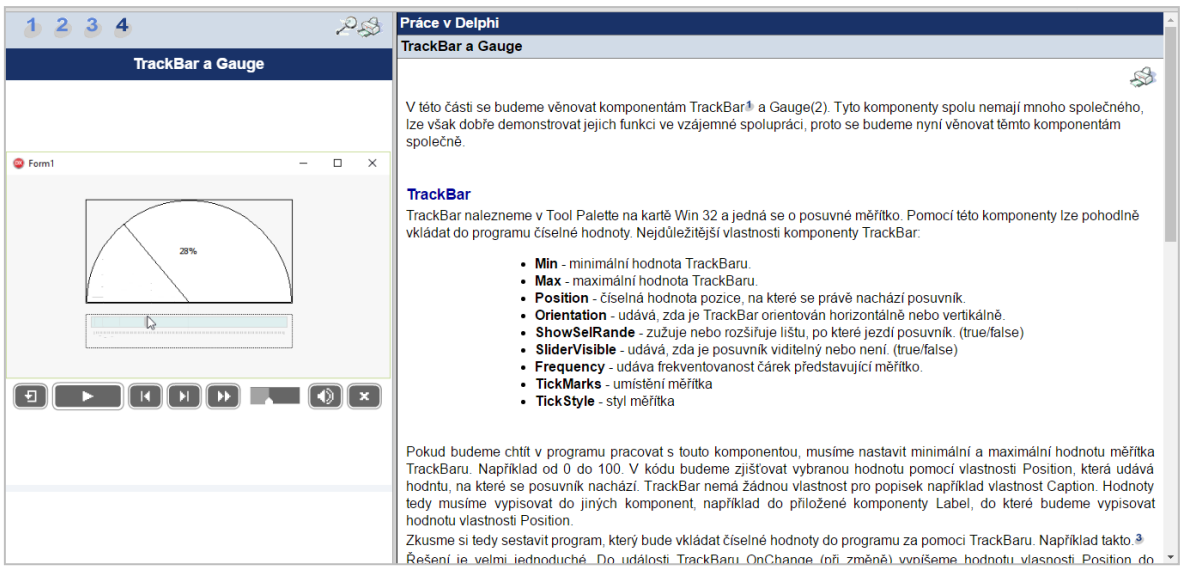
6.16 STUDIJNÍ ČLÁNEK TRACKBAR A GAUGE

Tento studijní článek pojednává o komponentech TrackBar a Gauge. Jedná se o posuvné měřítko a komponentu prezentující nějaký vývoj. Tyto komponenty nemají spolu mnoho společného, lze však dobře demonstrovat jejich funkci a vlastnosti ve vzájemné spolupráci. Tyto komponenty jsou zcela odlišné od komponent, se kterými se student dosavad setkal, a proto jim je věnován celý studijní článek.

Cílem tohoto studijního článku, je seznámit studenty s komponenty TrackBar a Gauge. Student pochopí význam těchto komponent a naučí se pracovat s jejich nejdůležitějšími vlastnostmi.

V tomto článku jsou tyto dvě komponenty popsány a jsou vysvětleny jejich nejpodstatnější vlastnosti. Vzhled a funkce těchto komponent, je k vidění v demonstrační části kódu.

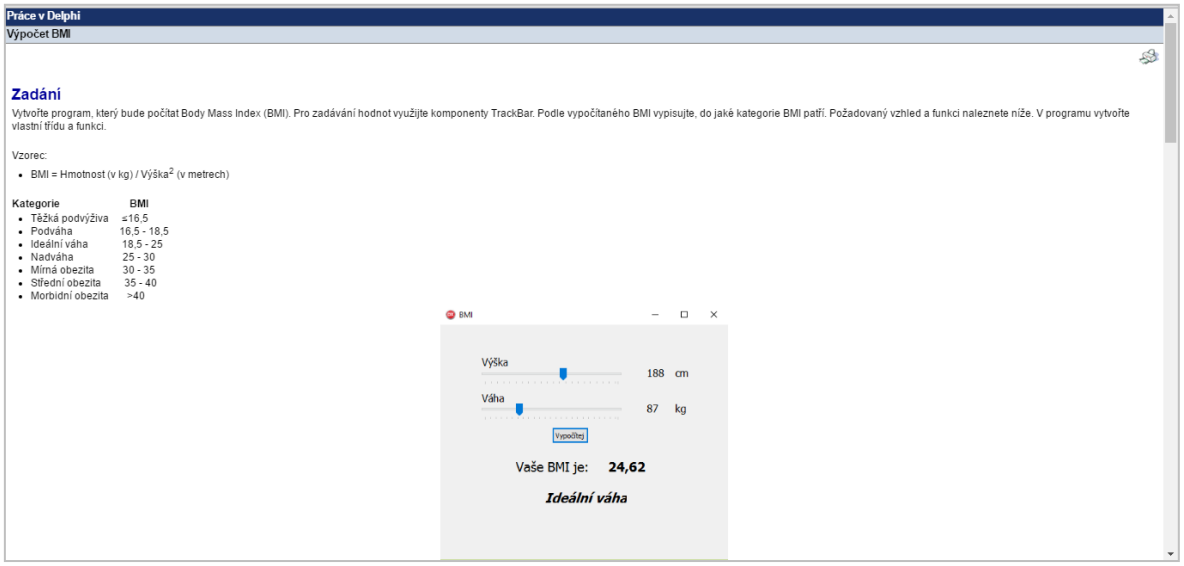
Na závěr tohoto studijního článku, je demonstrována vzájemná spolupráce těchto dvou komponent na příkladu. Tento příklad si student zkusí vypracovat společně se studijním článkem. Funkce programu s těmito komponenty je opět k vidění v demonstrační části kurzu. K dispozici je i hotové řešení, které lze stáhnout.



Obrázek 31: Studijní článek TrackBar a Gauge

6.17 CVIČENÍ VÝPOČET BMI

V tomto cvičení, má student za úkol vytvořit program, pro výpočet Body Mass Indexu (BMI), při kterém využije novou komponentu TrackBar. Pomocí této komponenty bude zadávat vstupní hodnoty, představující váhu a výšku člověka. Podle vypočítané hodnoty se vypíše kategorie, do které hodnota zapadá. V zadání je dále po studentovi požadováno vytvoření vlastní třídy, která bude zajišťovat výpočet. Student má k dispozici vzorec pro výpočet BMI, kategorie BMI, obrázek představující vzhled programu a animaci, představující funkci programu. Cílem tohoto cvičení, je osvojení si práce s vlastnostmi komponenty TrackBar.



Obrázek 32: Cvičení Výpočet BMI – zadání

Prvním krokem student vytvoří vzhled z komponent Label, Button a dvou nových komponent TrackBar. U komponent TrackBar vhodně nastaví příslušné vlastnosti. Ještě než se studen pustí do kódu, měl by zajistit vypisování hodnot komponent TrackBar, jelikož tato komponenta nemá k dispozici žádný popisek, který by ukazoval svou aktuální hodnotu. Toto vypisování student zajistí v události OnChange komponenty TrackBar, kde bude aktuální hodnotu vypisovat do komponenty Label.

Nyní student začne pracovat na zdrojovém kódu. Měl by začít vlastní třídou, která bude obsahovat potřebné atributy a metodu, která bude počítat výsledek. Pro univerzálnost by se hodila funkce s parametry pro váhu a výšku. Při definici této funkce by si student měl dát pozor, v jakých jednotkách počítá, je totiž třeba počítat hodnotu pro výšku v metrech.

Po vytvoření funkce je třeba výpočet spustit v události OnClick komponenty Button. Zde se bude volat funkce pro výpočet, do jejíchž parametrů se budou vkládat hodnoty nastavené v komponentech TrackBar. Výsledek se vypíše do příslušné komponenty Label. Nakonec se musí zajistit vypisování kategorií. To student provede pomocí podmínek. Samozřejmostí je vytvoření a zrušení objektů ve vhodných událostech.

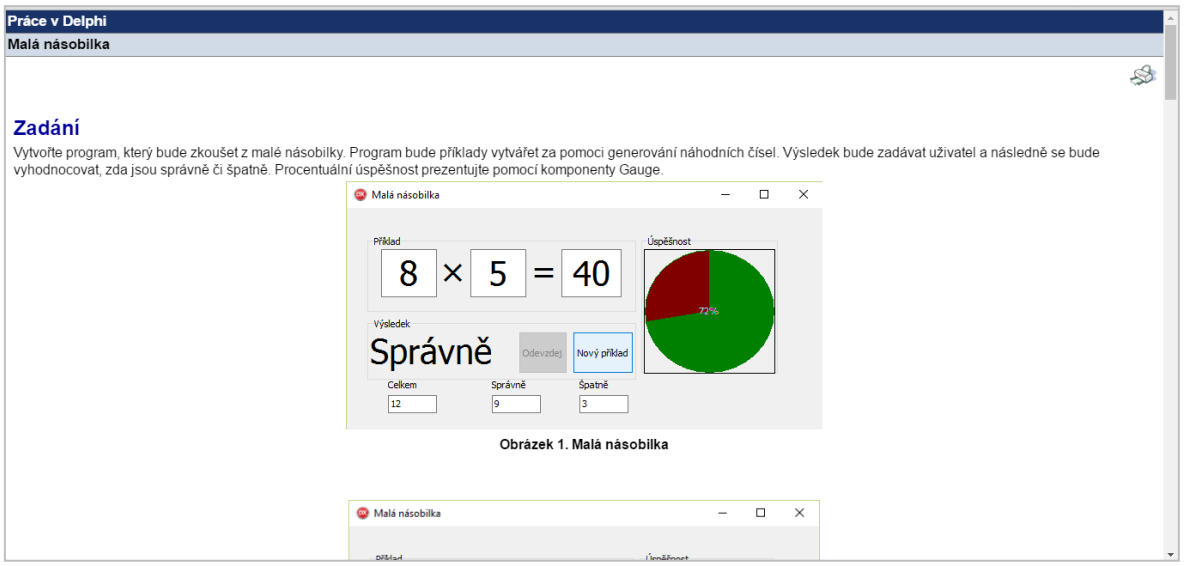
Opět má student k dispozici návrh řešení, ve kterých jsou jednotlivé kroky doplněny ukázkou částí zdrojového kódu. Na konci návrhu řešení lze stáhnout hotový projekt.

Obrázek 33: Cvičení Výpočet BMI - návrh řešení

6.18 CVIČENÍ MALÁ NÁSOBILKA

Student v tomto cvičení má za úkol vytvořit program, který bude zkoušet uživatele z malé násobilky. Tento program bude náhodně vytvářet příklady, vyhodnocovat správnost odpovědi, počítat počet správných a špatně zodpovězených příkladů a na závěr graficky znázorňovat procentuální úspěšnost uživatele. Student se dále seznámí s novou komponentou GroupBox, která komponenty třídí do skupin a dává vzhledu programu řád. Pod zadáním je ukázka požadovaného vzhledu programu. Funkce programu je popsána v animaci pod ukázkou vzhledu. Na konci zadání jsou typy pro řešení, je tam nápověda, jak generovat náhodná čísla.

Cílem tohoto cvičení, je utvrzení znalostí a dovedností žáka při práci s komponenty jako je Button, Edit, Label nebo Gauge. Dále si student procvičí práci s vlastností Enabled, která povoluje přístup ke komponentě. Student si v tomto cvičení procvičí logické myšlení. Posledním cílem je seznámit studenty s novou komponentou GroupBox.



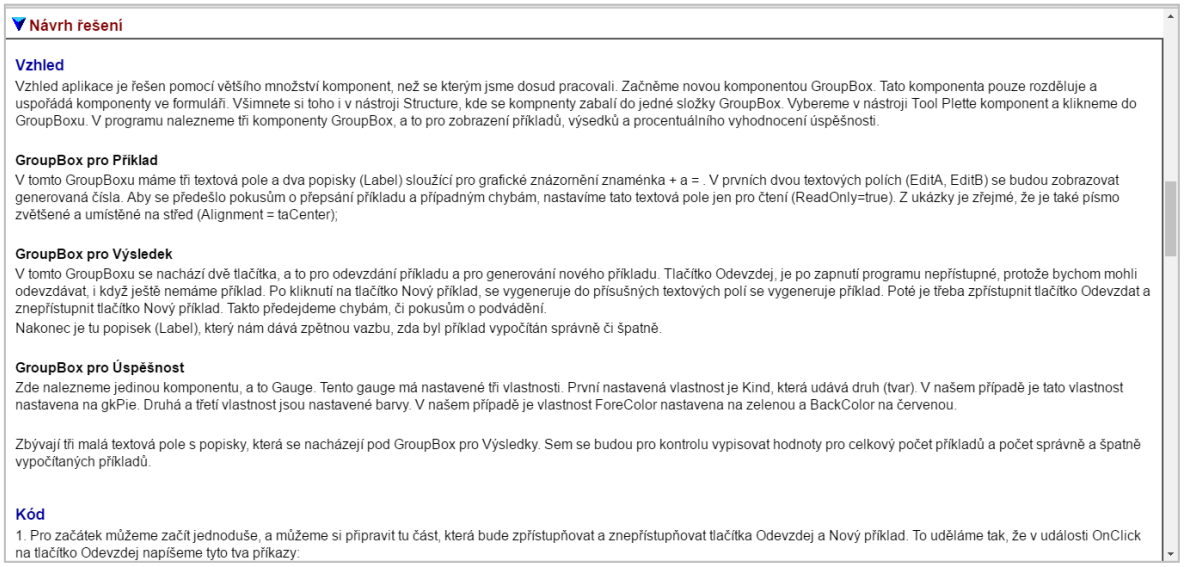
Obrázek 34: Cvičení Malá násobilka – zadání

Při tvorbě vzhledu, bude mít student více práce, než kdy dosud měl. Nejprve musí vložit do formuláře komponenty GroupBox. U těchto komponent změní vlastnost Caption, která obsahuje popisek komponenty. Dále vloží do komponent GroupBox další příslušné komponenty podle ukázky vzhledu, která je k dispozici pod zadáním. U příslušných komponent Edit a Button student nastaví jejich vlastnost na Enabled na False. Tím danou komponentu znepřístupní. Tyto komponenty jsou znepřístupněny pro znemožnění uživateli přepisovat vygenerovaný příklad, nebo odeslat odpověď ve chvíli, kdy se ještě nevygeneroval příklad. Jako poslední krok při tvorbě vzhledu, student nastaví komponentu Gauge. U této komponenty nastaví druh (vlastnost King) a barvy.

Po vytvoření vzhledu se student zaměří na zdrojový kód. Ze zadání není po studentovi požadováno vytvoření vlastní třídy, jelikož se předpokládá, že student v této fázi kurzu vytvoření třídy bere za samozřejmost. Samozřejmě není nutné vytvářet vlastní třídu, která bude obstarávat hlavní funkci programu, vše by šlo řešit pouze v událostech. Nicméně by toto řešení bylo velmi nepraktické. Studentova třída by měla obsahovat potřebné atributy a minimálně jednu proceduru pro práci s příkladem. Tato třída bude generovat dvě čísla, ze kterých vytvoří příklad. Tato čísla vypíše do příslušných textových polí a příklad vypočte pro kontrolu výsledku. Toto generování příkladu se bude dít v události OnClick tlačítka „Nový příklad“. V události OnClick tlačítka „Odevzdej“ se bude kontrolovat správnost uživatelovi odpovědi. Dále se bude počítat celkový počet příkladů a z toho počet správných a špatných odpovědí. Nakonec se bude počítat procentuální úspěšnost

uživatele, kterou bude prezentovat za pomoci komponenty Gauge. V některých případech výpočtu může nastat, že ačkoli uživatel vypočte vše dobře, zobrazí se mu úspěšnost 99 %. To může nastat z důvodu vady použitého vzorce. Pro tento případ může student využít zaokrouhlování výsledku nahoru. Vytvoření a zrušení objektů v příslušných událostech je samozřejmostí.

Student má k dispozici i návrh řešení, ve kterém je tvorba vzhledu a zdrojového kódu popsána a vysvětlena krok za krokem. K jednotlivým krokům tvorby zdrojového kódu jsou k dispozici ukázky částí zdrojových kódů. Na konci je k dispozici i soubor s hotovým řešením, které si student může stáhnout.


Návrh řešení

Vzhled

Vzhled aplikace je řešen pomocí většího množství komponent, než se kterým jsme dosud pracovali. Začněme novou komponentou GroupBox. Tato komponenta pouze rozděluje a uspořádá komponenty ve formuláři. Všimnete si toho i v nástroji Structure, kde se kompnenty zabalí do jedné složky GroupBox. Vybereme v nástroji Tool Plette komponent a klikneme do GroupBoxu. V programu nalezneme tři komponenty GroupBox, a to pro zobrazení příkladů, výsedků a procentuálního vyhodnocení úspěšnosti.

GroupBox pro Příklad

V tomto GroupBoxu máme tři textová pole a dva popisky (Label) sloužící pro grafické znázornění znaménka + a = . V prvních dvou textových polích (EditA, EditB) se budou zobrazovat generovaná čísla. Aby se předešlo pokusům o přepsání příkladu a případným chybám, nastavíme tato textová pole jen pro čtení (ReadOnly=true). Z ukázky je zřejmé, že je také písmo zvětšené a umístěné na střed (Alignment = taCenter);

GroupBox pro Výsledek

V tomto GroupBoxu se nachází dvě tlačítka, a to pro odevzdání příkladu a pro generování nového příkladu. Tlačítko Odevzdej, je po zapnutí programu nepřístupné, protože bychom mohli odevzdávat, i když ještě nemáme příklad. Po kliknutí na tlačítko Nový příklad, se vygeneruje do přísušných textových polí se vygeneruje příklad. Poté je třeba zpřístupnit tlačítko Odevzdat a znepřístupnit tlačítko Nový příklad. Takto předejdeme chybám, či pokusům o podvádění.
Nakonec je tu popisek (Label), který nám dává zpětnou vazbu, zda byl příklad vypočítán správně či špatně.

GroupBox pro Úspěšnost

Zde nalezneme jedinou komponentu, a to Gauge. Tento gauge má nastavené tři vlastnosti. První nastavená vlastnost je Kind, která udává druh (tvar). V našem případě je tato vlastnost nastavena na gkPie. Druhá a třetí vlastnost jsou nastavené barvy. V našem případě je vlastnost ForeColor nastavena na zelenou a BackColor na červenou.

Zbývají tři malá textová pole s popisky, která se nacházejí pod GroupBox pro Výsledky. Sem se budou pro kontrolu vypisovat hodnoty pro celkový počet příkladů a počet správně a špatně vypočítaných příkladů.

Kód

1. Pro začátek můžeme začít jednoduše, a můžeme si připravit tu část, která bude zpřístupňovat a znepřístupňovat tlačítka Odevzdej a Nový příklad. To uděláme tak, že v události OnClick na tlačítko Odevzdej napíšeme tyto tva příkazy:

Obrázek 35: Cvičení Malá násobilka - návrh řešení

6.19 ÚKOL NÁKLADY NA JEDNOHO CESTUJÍCÍHO

Tento úkol si klade za cíl jedinou věc, a to prověřit studentovi schopnosti práce s komponenty. Student má za úkol vytvořit program, který bude počítat náklady na jednoho cestujícího, a to při cestování automobilem a autobusem. Cenu nákladů na jednoho člena se vypočítá ze spotřeby paliva, ceny paliva, počtu cestujících, délky trasy ceny mýtného a dalších poplatků. U autobusu se bude zohledňovat i cena personálu a pro porovnání se zadá i cena jedné jízdenky. V zadání se po studentovi požaduje vytvoření dvou tříd, které budou představovat automobil a autobus. Student by měl sám přijít na možnost řešení těchto tříd dědičností. Dále student v zadání má k dispozici obecné informace o automobilu a autobusu, podle kterých by měl zohlednit nastavení některých komponent v programu. Opět je k dispozici ukázka vzhledu a funkce programu za pomoci

obrázku a animace. Student má i k dispozici soubor ke stažení, ve kterém je spustitelný soubor hotového programu. Tak může porovnat funkčnost svého programu s hotovým programem. V typech pro řešení nalezne student radu, ohledně přejmenování důležitých komponent pro lepší orientaci při tvorbě programu.

K tomuto úkolu jsou i vypsány pravidla hodnocení. Zohledňuje se celková funkčnost programu (2 body), úhlednost zdrojového kódu (2 body), způsob použití funkcí (3 body), studentovu práci s komponenty (3 body) a správnost vypočítaných hodnot (1 bod). Celkem tedy může student získat 11 bodů.

Student je tedy seznámen s požadavky na program a ve výsledku veškerá složitost tohoto programu, je v práci s komponenty, neboť je jich v tomto programu opravdu mnoho. Vzhled se skládá z komponent GroupBox, Edit, Label a TrackBar.

Ideálním postupem studenta, je kompletně vytvořit a nastavit komponentu GroupBox pro automobil. Z tohoto GroupBoxu vytvoříme kopii, která bude sloužit pro Autobus. Tím student jen komponenty přenastaví a přidá další potřebné komponenty pro zadávání ceny personálu a ceny jízdenky. Poté přidá tlačítko pro výpočet. Student nakonec vytvoří GroupBox pro výsledky automobilu a GroupBox pro výsledky autobusu.

Ve zdrojovém kódu si student ideálně vytvoří vlastní třídu pro automobil. Tato třída bude obsahovat potřebné atributy a funkci pro výpočet. Nejuniverzálnějším řešením by bylo, kdyby tato funkce obsahovala parametry pro veškeré potřebné hodnoty. V definici funkce se rovnou do výstupu funkce vypočítá výsledek, který bude představovat celkové náklady na jednu cestu. Poté si student vytvoří třídu autobus, která bude potomkem třídy automobil. Pro výpočet celkových nákladů na jednu cestu u autobusu je třeba zohlednit i cenu personálu. Proto tuhle funkci předefinujeme. Vytvoření a zrušení objektu Automobil a Autobus ve vhodných událostí je samozřejmostí. Poté je také důležité nastavit události OnChange všech komponent TrackBar, pro vypisování jejich aktuálních hodnot do příslušných komponent label. Nakonec student vytvoří událost OnClick jediného tlačítka. V této události se získají hodnoty z příslušných komponent a vloží se funkcí. Výsledky těchto funkcí se vypíší do příslušných komponent. Cena na jednoho člena se zjistí vydělením výsledku celkových hodnot počtem cestujících. Výsledek vypíšeme do příslušných komponent.

Práce v Delphi

Náklady na jednoho cestujícího

Zadání

Vytvořte kalkulační program, který bude počítat cenu nákladů na cestu na jednoho pasažéra, a to u automobilu a autobusu. Náklady se budou počítat z následujících údajů:

- Spotřeba paliva
- Cena paliva
- Počet cestujících
- Délka trasy
- Ceny mýtného
- Další poplatky

U Autobusu ještě přihlédneme k těmto údajům:

- Cena personálu
- Cena jízdenky

V programu použijte dvě třídy. Jednu pro automobil a druhou rozšířenou pro autobus. Ve třídách budou příslušné atributy a funkce, která bude řešit výpočet.

Obecné informace Automobil:

Běžná spotřeba: 5 - 10 l/100km

Cena za palivo: dle aktuální ceny

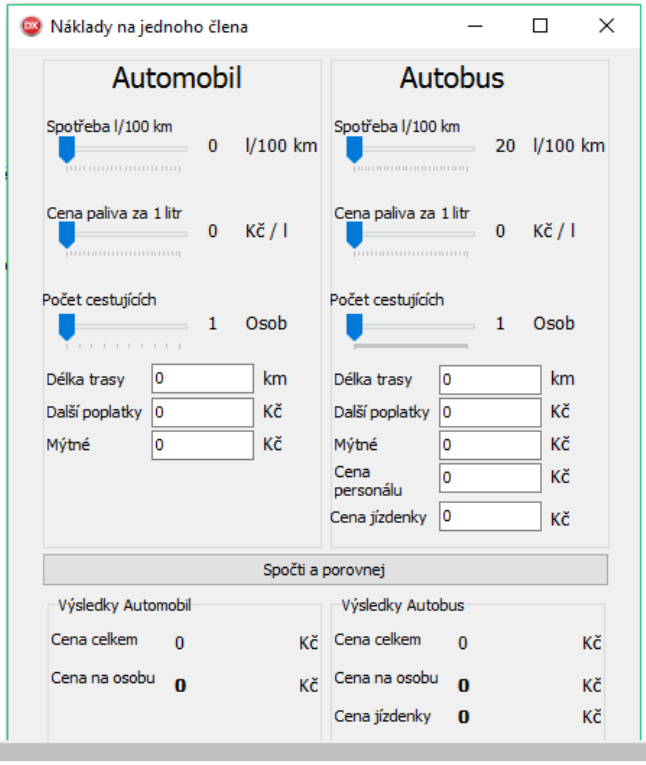
Počet cestujících: 1 - 7 (u dodávek i více)

Obecné informace Autobus:

Běžná spotřeba: cca 40 l/100km a i více (lze pouze typovat)

Cena za palivo: dle aktuální ceny (cena nafty)

Počet cestujících: 1 - 55 (v některých případech 60 i více)

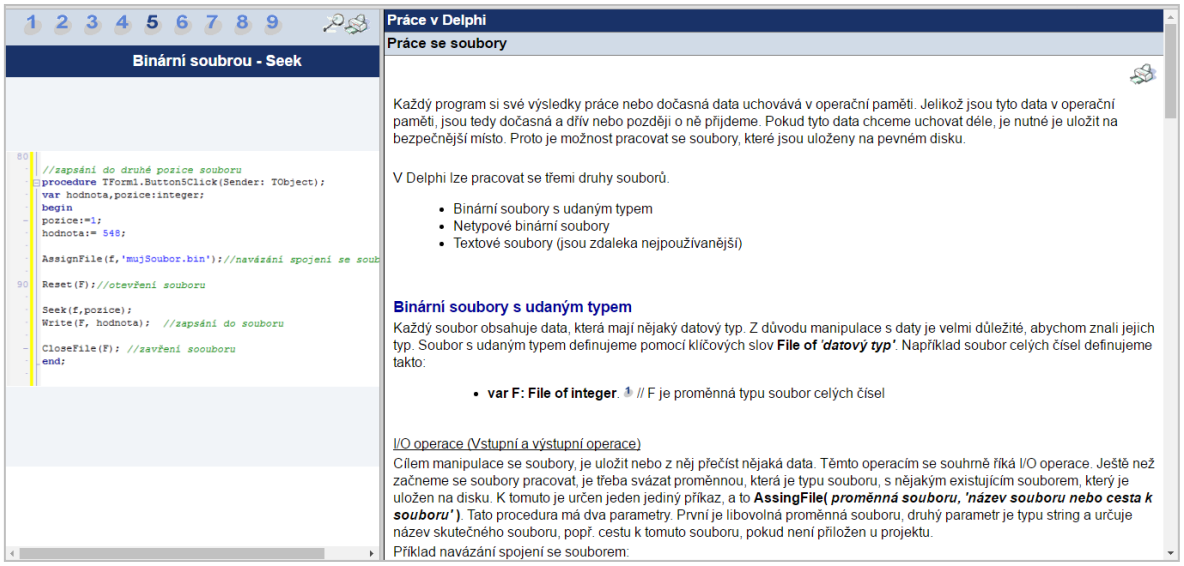


Obrázek 36: Úkol Náklady na jednoho člena

6.20 STUDIJNÍ ČLÁNEK PRÁCE SE SOUBORY

Tento článek pojednává o práci s binárními a textovými soubory v jazyce Object Pacal. Předpokládá se, že se studenti již s prací se soubory již setkali, a to v předmětu Programování 1. Nicméně je toto téma do kurzu zařazeno, jelikož se nepředpokládá dostatečná zkušenost studenta s touto problematikou, a také se jedná o důležité téma, které je třeba zvládnout. Cílem tohoto studijního článku, je seznámit studenta s prací se soubory v Object Pascalu. Student by tímto článkem měl pochopit rozdíl mezi binárními soubory a textovými soubory. Student se také seznámí s příkazy pro práci se soubory.

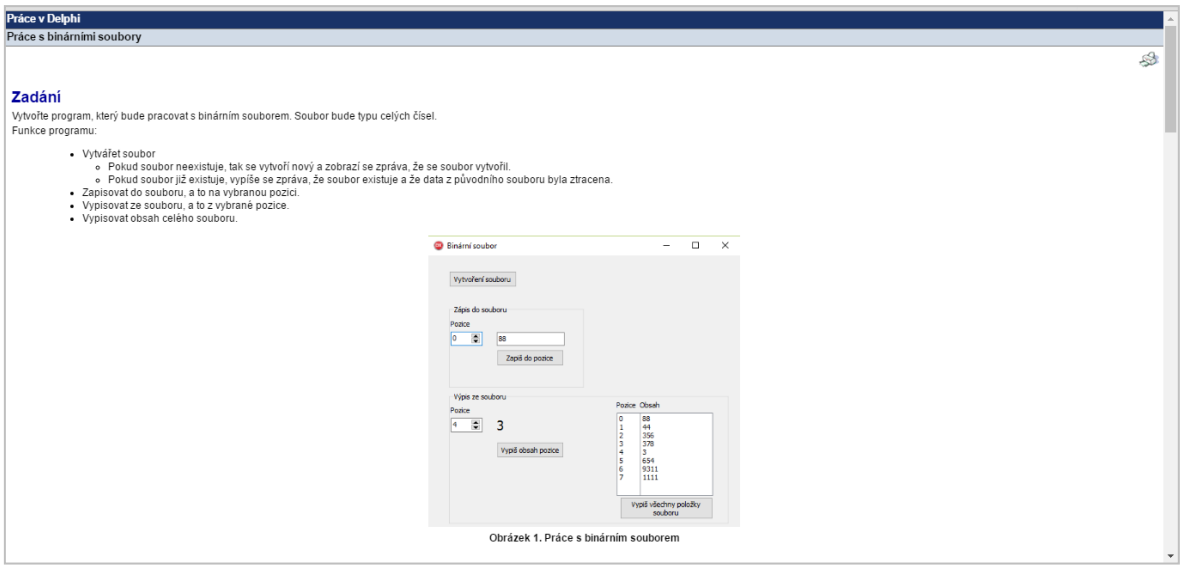
Jak již bylo zmíněno, student se v tomto článku seznámí jak s binárními, tak s textovými soubory. Nalezne zde vysvětlení příkazů pro práci se soubory, které jsou doprovázeny ukázkami zdrojových kódu v demonstrační části kurzu.



Obrázek 37: Studijní článek Práce se soubory

6.21 CVIČENÍ PRÁCE S BINÁRNÍMI SOUBORY

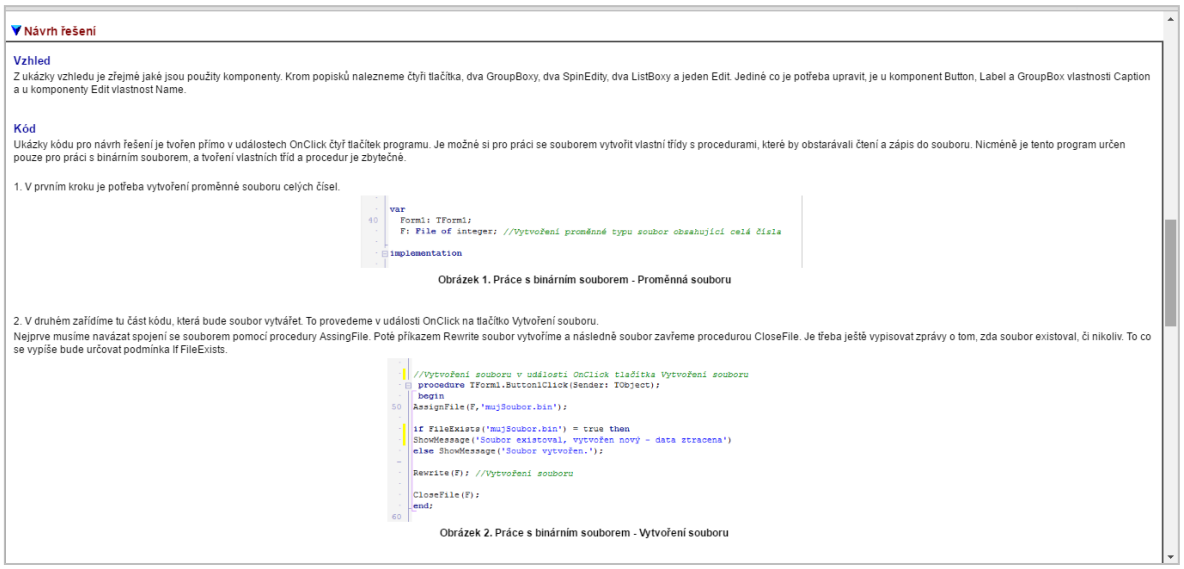
V tomto cvičení si student utvrdí znalosti nabyté v předchozím studijním článku. Úkolem studenta, je sestavit program, který bude binární soubor vytvářet, zapisovat vybrané pozice souboru, vypisovat z vybrané pozice souboru a vypisovat obsah celého souboru. Student má k dispozici i obrazovou ukázku vzhledu programu, a funkci programu, která je znázorněna animací.



Obrázek 38: Cvičení Práce s binárními soubory – zadání

Student prvním krokem vytvoří vzhled programu. Jsou použity dvě komponenty GroupBox, které rozdělují komponenty pro zápis a výpis do souboru. Zápis do pozice v souboru, je prováděn za pomoci komponenty SpinEdit (číslo pozice), Edit (obsah který chceme zapsat) a Button (zapíše do pozice). Pro výpis ze souboru jsou použity komponenty SpinEdit (číslo pozice), Label (pro vypsání obsahu z pozice) a Button (spuštění výpisu). Vypisovat budeme i celý obsah souboru, a to do komponenty ListBox. V ukázce programu je i druhý ListBox, který vypisuje číslo pozice. Nakonec je v programu tlačítko, kterým vytváříme nový soubor. Po vytvoření vzhledu se student zaměří na události OnClick čtyř tlačítek. V těchto událostech využije příkazů z předchozího studijního článku.

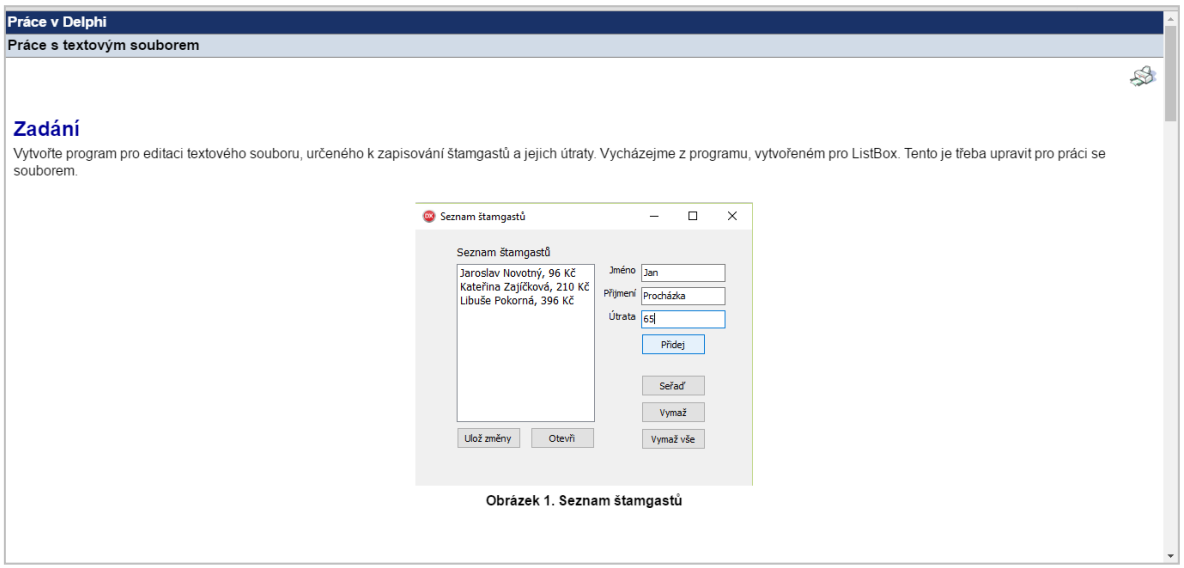
Ke konci cvičení je k dispozici návrh řešení, kde student nalezne popis a vysvětlení každého kroku. Společně s popisem nalezne i obrazovou ukázku zdrojového kódu. Na konci řešení je k dispozici i soubor s hotovým řešením, který si student může stáhnout a otevřít ve svém vývojovém prostředí.



Obrázek 39: Cvičení práce s binárními soubory – řešení

6.22 CVIČENÍ PRÁCE S TEXTOVÝM SOUBOREM

Cílem tohoto cvičení, je utvrzení nabytých znalostí ze studijního článku Práce se soubory. V tomto programu student rozšíří svůj program, který vytvářel ve studijním článku o komponentě ListBox. Tento program bude do komponenty ListBox položky zapisovat, řadit od A-Z, mazat položky nebo smaže všechny položky z této komponenty. Obsah komponenty ListBox půjde uložit do souboru nebo ze souboru otevřít. K zadání je opět k dispozici ukázka vzhledu a funkce programu.

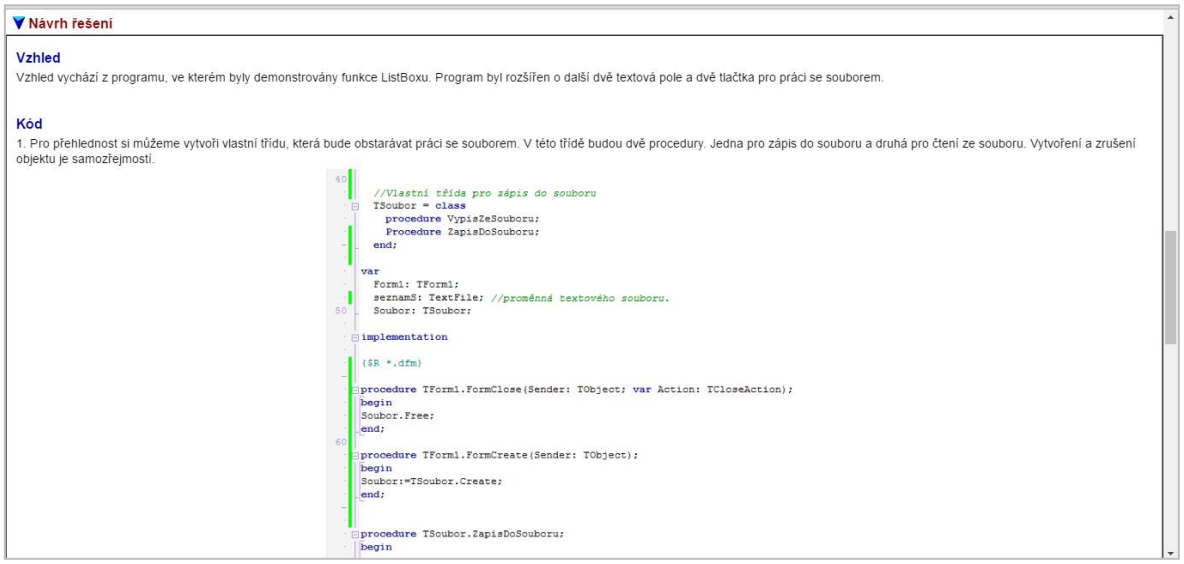


Obrázek 40: Cvičení Práce s textovým souborem – zadání

Student vytvoří vzhled, bez větších uprav vlastností komponent v nástroji Object Inspector. Poté ve zdrojovém kódu zařídí funkci programu. Je na studentovi, zda využije

vlastních funkcí nebo ne. Řešení programu je jednoduché, jelikož celou funkci programu zajistí pár příkazů přímo v událostech OnClick komponent Button.

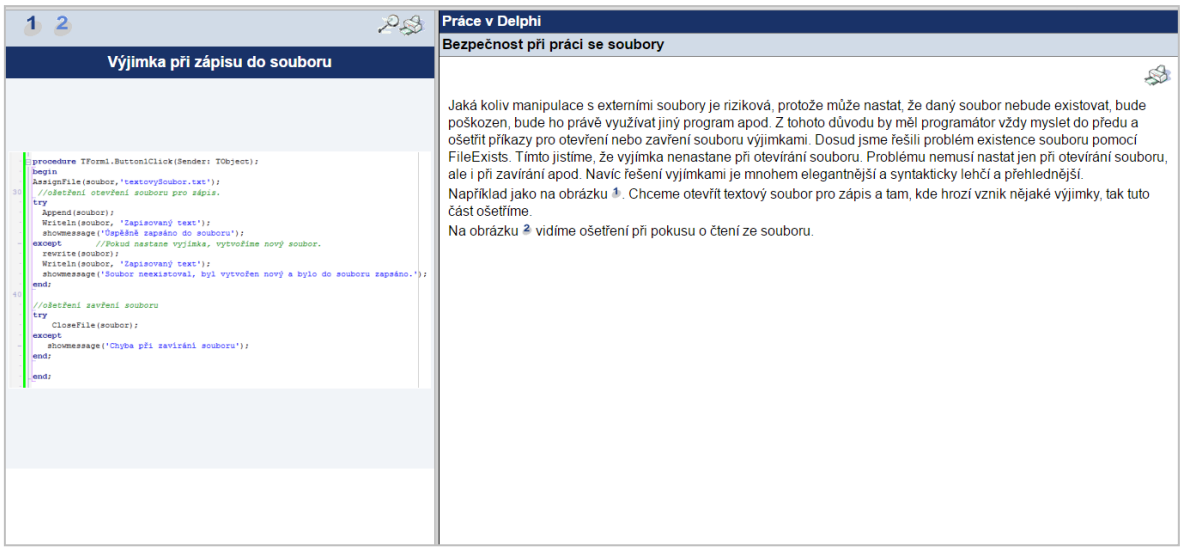
Ke konci cvičení je opět k dispozici návrh řešení, ve kterém je každý krok popsán a vysvětlen společně s obrazovou ukázkou zdrojového kódu. Ke konci cvičení lze opět stáhnout soubor s hotovým řešením.



Obrázek 41: Cvičení Práce s textovým souborem - návrh řešení

6.23 STUDIJNÍ ČLÁNEK BEZPEČNOST PŘI PRÁCI SE SOUBORY

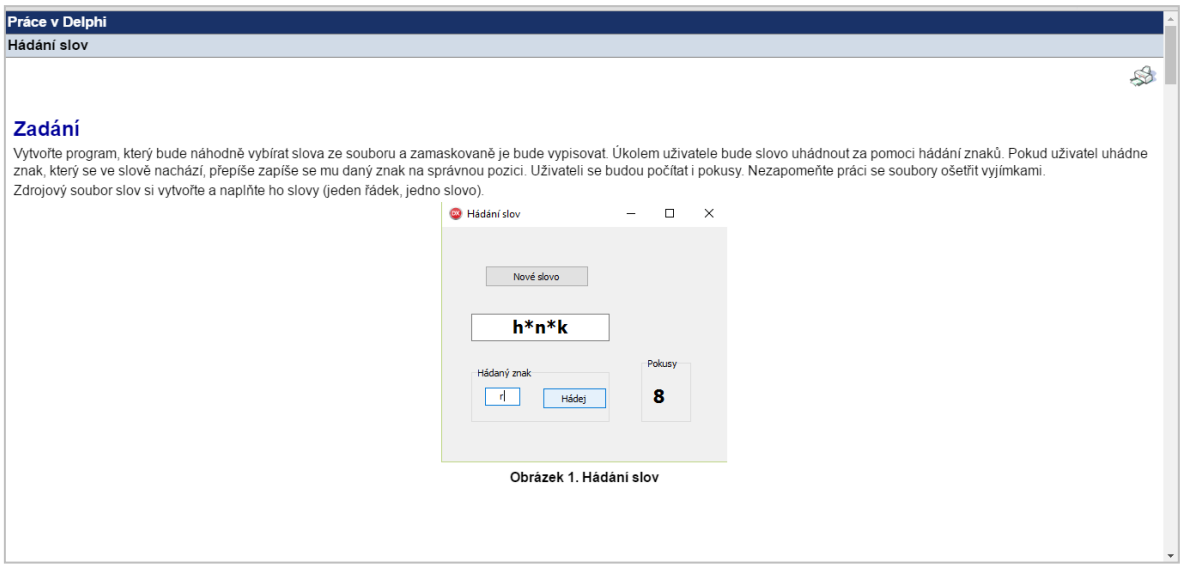
Tento studijní článek doplňuje článek o práci se soubory. Cílem tohoto studijního článku, je studenta seznámit s možností ošetření krizových míst ve zdrojovém kódu, za pomoci výjimek. Jelikož práce s externím souborem obnáší vždy riziko, student by měl práci se souborem ošetřit výjimkami. Ukázku využití výjimek student nalezne v demonstrační části studijního článku.



Obrázek 42: Studijní článek Bezpečnost při práci se soubory

6.24 CVIČENÍ HÁDÁNÍ SLOV

Cílem tohoto cvičení, je utvrzení schopností a znalostí, které student nabyl v tomto kurzu. Úkolem toho cvičení, je vytvořit program, který bude náhodně vybírat slova uložena v textovém souboru. Znaky tohoto slova zamaskuje samými hvězdičkami. Uživatel se pokusí co nejméně pokusy slovo uhádnout. Slovo uživatel hádá po znacích. Uhodne-li uživatel znak, který se ve slově nachází, zobrazí se na správné pozici. Jakmile Uživatel uhodně všechny znaky, vypíše se zpráva o počtu uživatelových pokusů. V zadání se po studentovi požaduje ošetřit práci se souborem výjimkami a vytvoření textového souboru se slovy, který student vytvoří mimo program. Pod zadáním se opět nachází ukázka vzhledu a funkce programu. Pro studenta jsou vypsány i typy pro řešení, ve kterých jsou rady pro řešení programu.



Obrázek 1. Hádání slov

Obrázek 43: Cvičení Hádání slov - zadání

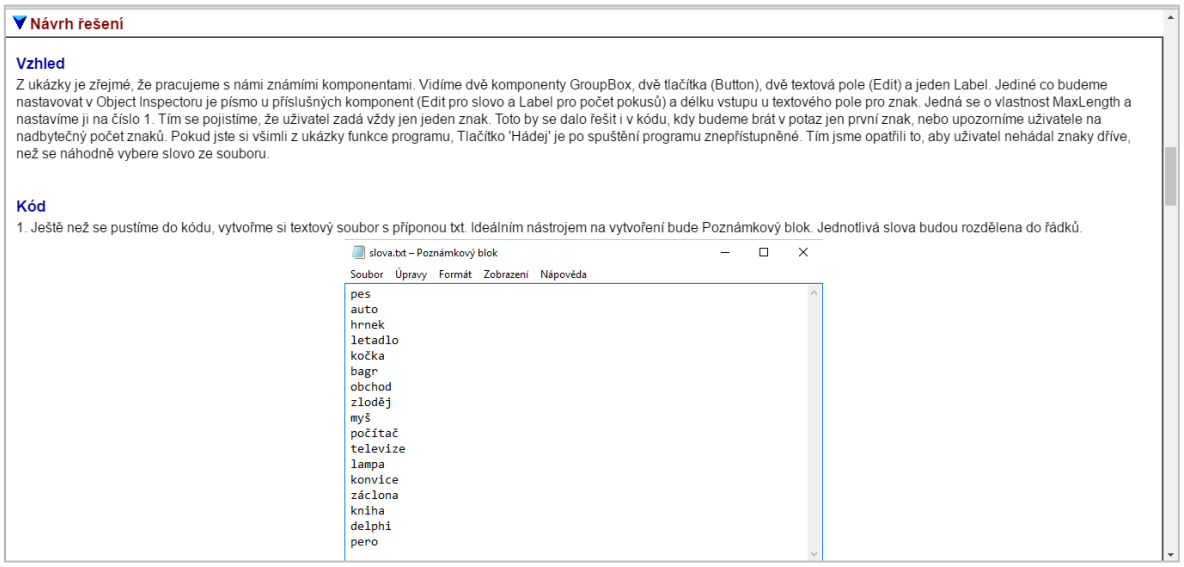
Student svou práci zahájí tvorbou vzhledu programu. Tento vzhled je jednoduchý a nemělo by být problémem tento vzhled sestavit. Vzhled je složen ze známých komponent GroupBox, Edit a Button. Ještě než se student pustí do zdrojového kódu, vytvoří si textový soubor, který naplní slovy, která se budou náhodně vybírat. To student udělá nejlépe v poznámkovém bloku.

Ve zdrojovém kódu si student ideálně vytvoří třídu, která bude mít za úkol slova ze souboru náhodně vybírat a následně slovo maskovat. Tyto dvě činnosti budou obstarávat dvě funkce. Funkce pro maskování slova bude ideálně obsahovat i parametr, pro slovo, které chceme zamaskovat.

Funkce, která bude slovo náhodně vybírat ze souboru, bude obsahovat dva cykly. První cyklus bude zjišťovat počet řádků v souboru. Následně se vygeneruje náhodné slovo od 0 do počtu řádků v souboru. Náhodné číslo bude představovat řádek vybraného čísla. Na základě vybraného čísla se v druhém cyklu vybere slovo. Toto slovo se bude vkládat do druhé funkce, která za pomoci cyklu slovo zamaskuje.

Poté student vytvoří událost na tlačítko „Hádej“. V této události se bude prověřovat shoda znaku se znaky vybraného slova. Pokud se najde shoda, tak se cyklem změní znak na dané pozici v maskovaném slově, a toto slovo se vypíše. Toto se bude opakovat, dokud uživatel slovo neuhádne. Po celou dobu se počítají pokusy, které se vypisují uživateli.

Pod zadáním je připraven i návrh řešení, které vysvětluje navrhnutý postup krok za krokem společně s ukázkami zdrojového kódu. Na konci návrhu řešení je opět k dispozici soubor s hotovým řešením, které lze stáhnout a spustit ve vývojovém prostředí Delphi.



Obrázek 44: Cvičení Hádání slov - návrh řešení

6.25 ÚKOL HRA S ČÍSLY

Tento úkol si klade za cíl prověřit schopnosti a znalosti studenta, které během tohoto kurzu nabyl. Student by měl zvládnout práci s komponenty, jejich vlastnostmi a událostmi. Dále by měl zvládnout pracovat s třídami, objekty a s jejich atributy a metodami. Měl by zvládnout i práci se souborem, se kterým se v tomto programu také pracuje. Posledním cílem tohoto cvičení je, prověřit, zda student plně pochopil principy práce v Delphi. To se prověří nejen zvládnutím již probrané problematiky, ale také zvládnutím použití nových věcí, se kterými se student dosud nesetkal. K těmto novým věcem by studentovi měla stačit jen nápověda, kterou nalezne v typech pro řešení.

Student v této části kurzu má za úkol vytvořit program, který bude představovat hru s čísly. Tento program bude generovat tři čísla, a to od 0 do 2. Na základě kombinace těchto čísel, se body uživatele budou přičítat, odčítat, nebo zůstanou beze změny. Aby uživatel mohl čísla generovat, musí si vsadit, a to od 1 do 10 bodů. Pokud vyjdou všechna čísla stejná, vsazené body se přičítají k celkovému počtu bodů. Pokud naopak všechna čísla budou jiná, vsazené body se od celkového počtu bodů odečítají. Poslední možností je, že dvě čísla budou stejná. V tomto případě zůstanou body beze změny. Uživatel má pět

pokusů, pomocí kterých by se měl pokusit dosáhnout co nejvyššího počtu bodů. Po vyčerpání všech pokusů se uživatelům výsledek uloží do souboru. Tento obsah souboru bude permanentně vypsán v komponentě ListBox.

Student má v zadání vypsaná pravidla hry, podle kterých by se měl program řídit. Dále jsou v zadání vypsané všechny funkce, které program bude mít. Následně je opět k dispozici ukázka vzhledu a funkce programu. Lze v zadání i stáhnout jen spustitelný soubor programu, pomocí kterého si student může zkontrolovat funkčnost svého programu. Nakonec zadání jsou i typy pro řešení programu. Zde student nalezne typy jak se vypořádat s novými záležitostmi. První z nich je práce s komponentou InputBox. Ta se volá přímo v programu a nepracuje se s ní při vytváření vzhledu, tak jak to je s jinými komponenty. Druhou novou záležitostí je práce s komponentou BitButton. Jedná se o komponentu Button, do které se dá vložit obrázek z přednastavených vzorů nebo i vlastní obrázek.

Prvním úkolem na řešení tohoto programu, je vytvoření vzhledu. Student bude pracovat s komponenty Edit, TrackBar, GroupBox, Button a ListBox. V tomto programu bude muset nastavit některé vlastnosti komponent Edit, Button a TrackBar tak, jak je to zřejmé z popisu funkce programu.

Ve zdrojovém kódu je na studentovi, zda uzná za vhodné využít třídu nebo metody, které by obstarávali funkce programu. V události komponenty BitButton se bude odehrávat veškerý běh programu. To znamená, že je třeba generovat tři čísla. Podle kombinace těchto čísel se následně bude rozhodovat, jak se vynaloží body uživatele. Budou se i kontrolovat pokusy. Jakmile uživateli pokusy dojdou, zapíše se jeho výsledek s jeho jménem do souboru. Obsah tohoto souboru bude permanentně vypisován do komponenty ListBox. Z ukázky funkce programu, je potřeba zařídit i zadávání jména uživatele. V ukázce je toto řešeno pomocí InputBoxu, který se zobrazí ihned při spuštění programu. To student provede v události formuláře OnCreate. Pokud bude uživatel chtít hrát znovu, je třeba komponentu InputBox zobrazit znovu, jelikož může jít o jiného hráče. Pro tutora je k dispozici soubor s hotovým řešením, které lze stáhnout a spustit ve vývojovém prostředí Delphi.

Práce v Delphi

Hra s čísly

Zadání

Vytvořte program Hra s čísly. Tento program bude představovat hru, ve které bude uživatel losovat tři náhodná čísla.

Pravidla hry

Každý hráč začíná s dvaceti body. Aby mohl hráč čísla losovat, musí vsadit od 1 do 10 bodů. V závislosti na kombinaci vylosovaných čísel, se vsazené body k celkovému počtu bodů buď odečítají, přičítají anebo celkový stav bodů zůstává nezměněn. Po vyčerpání pěti pokusů se výsledný počet bodů se jménem hráče vypíší do ListBoxu a obsah této kompnenty se ihnet uloží do souboru. Cílem hry, je vyhrát co nejvíce bodů.

Losování čísel

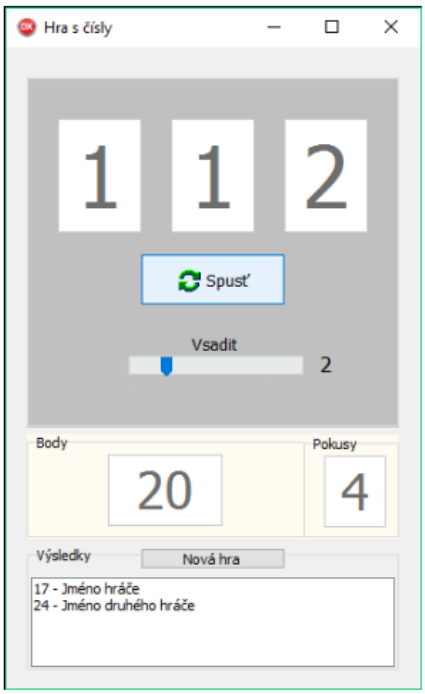
- Každé z čísel může nabýt hodnot 0, 1 nebo 2.

Kombinace čísel

- Všechyna čísla stejná = vsazené body se přičítají k celkovému počtu bodů.
- Dvě čísla stejná = celkový počet bodů zůstává nezměněn.
- Každé číslo jiné = vsazené body se odečítají od celkového počtu bodů.

Požadované funkce programu

- Losování třech náhodných čísel od 0 do 2.
- Počítání pokusů
- Přičítání nebo odečítání bodů v závisloti na kombinaci vylosovaných čísel.
- Nebude možné losovat, pokud není vsazeno.
- Nebude možné dále hrát, pokud hráč vyčerpat pokusy.
- Po skončení hry se uživatelovo jméno s výslednými body se vloží do ListBoxu a rovnou se uloží do souboru.
- Soubor s výsledky se do ListBoxu bude načítat při startu programu. Pokud soubor neexistuje, vytvoří se.
- Jméno hráče, se bude vkládat přes InputBox. (Více v typech pro řešení)
- Při nové hře se program bude opět ptát na jméno hráče pomocí InputBoxu. Bude přednastavené jméno minulého hráče.



Obrázek 1. Hra s čísly

Obrázek 45: Úkol Hra s čísly - zadání

6.26 ZÁVĚREČNÝ TEST

Tento závěrečný teoretický test si klade za cíl, prověřit studentovu znalost principů práce ve vývojovém prostředí Delphi, znalost práce s komponenty s jejich vlastnostmi a událostmi. Prověřuje studentovu orientaci ve zdrojovém kódu, znalost dobrých zásad programátora a znalost syntaxe. Student po odevzdání testu dostane zpětnou vazbu o správnosti jeho odpovědí. V tomto testu je připraveno deset otázek, které jsou tvořeny z otázek dichotomických (jedna správná odpověď ze dvou možných, například Ano-Ne), polytomických (jedna správná odpověď z více nabízených) a výčtových (umožňují více správných odpovědí). U některých otázek je poskytnuto i vysvětlení, pokud student odpoví na danou otázku chybně.

Obrázek 46: Závěrečný test

ZÁVĚR

V této diplomové práci byl vytvořen distanční kurz pro předmět Programování 2. Student do tohoto kurzu vstupuje se znalostmi programovacího jazyka Object Pascal, které nabyl v předmětu Programování 1. První s čím se student v kurzu setká, je kapitola o objektově orientovaném programování, ve které dožene všechny mezery ve vědomostech o OOP. Dále se student seznámí s vývojovým prostředím Delphi a s jeho nejdůležitějšími nástroji, které bude potřebovat pro svou další práci v kurzu. Poté následuje praktická část kurzu. Student v této části prakticky pracuje zároveň s kurzem i ve vývojovém prostředí Delphi. Pro začátek se seznámí s vývojovým prostředím a s jeho hlavními nástroji. Poté přejde k prvním programům, ve kterých si vyzkouší pracovat s komponenty, formulářem a s jejich vlastnostmi a událostmi. S těmito vlastnostmi se student naučí pracovat nejen v nástroji Object Inspector, ale také i ve zdrojovém kódu. Nakonec se student pustí do vytváření programů, ve kterých využívá vlastních tříd, atributů, metod a rysů OOP. Student se bude potýkat s mnoha problémy v podobně zadaných úkolu a cvičení, které bude muset vyřešit. Vyřešením těchto úkolů a cvičení student získá znalosti, zkušenosti a zdokonalí si své logické myšlení, které je třeba pro programování. V ideálním případě by student mohl samostatně některé programy zdokonalit nebo rozšířit o další funkce.

Tento kurz především dbal na podchycení základních principů práce ve VCL Forms Application. Myslím si, že by stálo za to, tento kurz rozšířit i o práci s komponenty jako například MainMenu, Timer, PageControl anebo práci s dialogy. Dále by kurz mohl obsahovat i možnosti práce s grafikou, práci s multimédii, DDL knihovny anebo podrobnější práci se soubory.

Ačkoli se příliš v Object Pascalu neprogramuje, lze velmi dobře využít pro výukové účely. Díky tomuto programovacímu jazyku student pochopí principy objektově orientovaného programování a nebude pro něj příliš velkým problémem přejít i na jiné objektově orientované programovací jazyky jako je například Java nebo C++. Věřím, že kurzu splní svůj hlavní záměr, a to seznámit studenta se základními principy práce v projektech VCL Forms Application. Student tedy získá základní znalosti a dovednosti, díky kterým bude schopen pokračovat ve svém zdokonalování i samostatně.

RESUMÉ

Cílem této kvalifikační práce, je vytvořit distanční kurz, pro předmět Programování 2. Student se v tomto distančním kurzu naučí základním principům práce ve vývojovém prostředí Delphi, a to v projektech VCL Forms Application. Nejprve se student v kurzu seznámí s rysy objektově orientovaného programování a se syntaxí v jazyce Object Pascal. Dále se seznámí s vývojovým prostředím a s jeho nejdůležitějšími nástroji, které student bude využívat při své další práci v kurzu. Dále následuje praktická část, kdy student bude zároveň s kurzem pracovat i ve vývojovém prostředí Delphi. Student si prakticky vyzkouší práci ve vývojovém prostředí Delphi, seznámí se s prací s komponenty, formulářem a také s jejich vlastnostmi a událostmi. Student se během kurzu setká s mnoha cvičeními a úkoly, podle kterých bude vytvářet dané programy. Tím student nabude nových znalostí a zkušeností. Student během tohoto kurzu získá základní znalosti a dovednosti, díky kterým bude schopen pokračovat ve svém zdokonalování i samostatně.

The aim of this theses is to create a distance learning course for subject Programming 2. Students in this distance learning course teaches the basic principles of work in the development environment Delphi in projects VCL Forms Application. First of students acquainted with the features of object-oriented programming and the syntax of Object Pascal. They also learn about the development environment and with its most important tools that students will use in their other coursework. This is followed by a practical part, the students will also work with the course and in the development environment Delphi. Students can virtually try working in a development environment Delphi, familiar with working with a component form and with their properties and events. Student during the course meets many exercises and tasks, according to which it will produce the programs. This student acquires new knowledge and experience. Student during the course acquire basic knowledge and skills thanks to which will be able to continue its improvement of its own.

SEZNAM LITERATURY

MARTON, Vojtěch. *Tvorba sady příkladů pro předmět Programování 2*. Plzeň, 2014. Bakalářská práce. ZČU, Fakulta pedagogická, Katedra výpočetní a didaktické techniky. Vedoucí práce Mgr. Tomáš Přibáň, Ph.D.

PÍSEK, Slavoj. *Delphi - začínáme programovat: podrobný průvodce začínajícího uživatele*. 2. upr. a aktualiz. vyd. Praha: Grada, 2002. ISBN 80-247-0547-8.

KEOGH, James Edward a Mario GIANNINI. *OOP bez předchozích znalostí: průvodce pro samouky*. Brno: Computer Press, 2006. ISBN 80-251-0973-9.

SEZNAM OBRÁZKŮ, TABULEK, GRAFŮ A DIAGRAMŮ

Seznam obrázků

PŘÍLOHY

Přílohy se nacházejí na přiloženém CD. Na tomto CD se nachází písemná část diplomové práce ve formátu PDF a docx. Dále obsahuje distanční kurz pro předmět Programování 2, a to v podobě kurzu a vyexportované offline verze eBook.

1. Diplomová práce_Vojtěch Marton.docx
2. Diplomová práce_Vojtěch Marton.pdf
3. Distanční kurz Programování 2_Projekt
4. Distanční kurz Programování 2_eBook