

IPAS Server – komunikační rozhraní



Obsah

Seznámení se systémem	4
Server IPAS.....	4
PHP knihovna clsIpas	4
.NET knihovna Ignum Kernel.....	4
Soubor RULE.XML.....	4
Komunikační protokol se serverem IPAS.....	5
Hlavička komunikace.....	5
Hlavička komunikace pro GZIP formát.....	5
Pravidla spojení.....	5
Testovací rozhraní.....	6
Zřízení testovacího účtu.....	6
Dostupná funkcionalita.....	6
Testování na produkčním systému.....	6
Popis PHP knihovny clsIpas.....	7
Přehled nejdůležitějších změn pro IPAS2.....	7
Popis jednotlivých tříd knihovny clsIpas.....	7
Popis třídy clsIpas.....	7
Instanční proměnné třídy.....	7
Metody třídy.....	8
Popis třídy IpasElementFunction.....	8
Funkčnost.....	8
Implementace.....	8
Příklad.....	9
Vzorový program.....	9
Vygenerovaný XML příkaz.....	9
Popis přidávaných funkcí ve třídě IpasAddFunction.....	9
Dynamické generování příkazů a knihovny clsIpas.....	10
Konfigurace generátoru knihovny.....	10
Princip tvorby knihovny a příkazů.....	11
Instalace knihovny.....	11
Volání příkazů serveru IPAS.....	11
Návratové hodnoty generátoru XML příkazů pro server IPAS.....	11
Server Ipas momentálně podporuje tyto příkazy přístupné pro partnery nebo subregistrátory.....	12
Podrobný popis důležitých příkazů.....	14
Příkaz Login.....	14
Příkaz GetDomainInfo.....	15
Příkaz CheckDomain.....	15
Příkaz Query.....	15
Příklady.....	15
Přihlášení k serveru IPAS.....	15
Volání příkazu IPAS GetDomainInfo.....	15
Volání příkazu IPAS CheckDomain.....	16
Volání příkazu IPAS Query.....	16
Volání příkazu OrderService pro vytvoření generického kontaktu.....	17
Volání příkazu OrderService pro registraci generické domény.....	18
Přehled změn v knihovně	18
Popis .NET knihovny IGNUK KERNEL.....	21
Popis třídy Ignum.Kernel.DirectSession.....	21
Příklady.....	21
Přihlášení k serveru IPAS	21

Volání příkazu IPAS GetDomainInfo.....	21
Volání příkazu IPAS Query.....	22
Přehled změn v knihovně	22
Obecný popis struktury XML příkazů.....	24
Příkazy pro zjišťování informací.....	24
Kontrola dostupnosti domény.....	24
Informace o doméně.....	24
Informace o kontaktu.....	26
Informace o virtuálním kontaktu.....	27
Načtení dat z databáze pomocí předdefinovaného dotazu.....	29
Ostatní domény.....	30
Registrace virtuálního kontaktu.....	30
Změna údajů virtuálního kontaktu.....	31
Registrace ostatních národních domén.....	31
Prodloužení ostatních národních domén.....	32
Popis odpovědí od serveru IPAS.....	32
Doména CZ a ENUM.....	33
Registrace kontaktu.....	33
Změna kontaktu.....	34
Smazání kontaktu.....	35
Registrace NSSETu.....	36
Změna NSSETu.....	36
Smazání NSSETu.....	37
Registrace domény CZ.....	38
Speciální registrace domény CZ.....	38
Prodloužení domény CZ a ENUM.....	38
Změna domény CZ a ENUM.....	38
Změna vlastníka domény CZ.....	39
Zrušení domény CZ a ENUM.....	40
Registrace domény ENUM.....	40
Změna vlastníka domény ENUM.....	41
Validace a revalidace vlastníka domény ENUM.....	41
Transfery kontaktů, NSSETů a domén CZ a ENUM.....	42
Zaslání Auth-ID pro transfer kontaktu.....	42
Zaslání Auth-ID pro transfer NSSETu.....	42
Zaslání Auth-ID pro transfer domény CZ a ENUM.....	42
Objednávka transferu kontaktu pod IGNUm.....	43
Objednávka transferu NSSETu pod IGNUm.....	43
Objednávka transferu domény CZ nebo ENUM pod IGNUm.....	43
Informace o kontaktu.....	44
Informace o NSSETu.....	44
Informace o doméně CZ a ENUM.....	44
Přidání NSSETu do seznamu uživatele.....	44
Odebrání NSSETu ze seznamu uživatele.....	44
Generické domény.....	45
Registrace generického kontaktu.....	45
Změna generického kontaktu.....	45
Registrace generické domény.....	46
Transfer generické domény.....	47
Prodloužení generické domény.....	47
Další příkazy.....	48
Přihlášení uživatele.....	48
Přílohy.....	49
Přehled nejdůležitějších dotazů pro příkaz Query.....	49

Seznámení se systémem

Server IPAS

Server IPAS je výkonným jádrem pro registraci CZ, generických a ostatních domén. V budoucnu se IPAS stane jádrem i pro registraci a správu webhostingů a dalších služeb poskytovaných společností Ignium. IPAS obsahuje nástroje pro poskytování služeb i jiných společností skrze společnost Ignium (např. partneři a subregistratori), kteří mohou se serverem libovolně komunikovat pomocí komunikačního rozhraní popsaného níže.

Server v současnosti obsahuje příkazy pro kompletní registraci CZ, generických a jiných domén. Dále pak příkazy pro ovládání fakturace, objednávání, kreditu a osobních nastavení uživatelů, správu uživatelských účtů.

PHP knihovna clsIpas

Knihovna clsIpas vznikla pro potřeby partnerů a subregistratorů společnosti Ignium komunikovat se serverem IPAS.

Vývojovým jazykem knihovny se stalo PHP 5, které je pro programování knihoven výrazně pokročilejší.

Tato knihovna vychází z interní knihovny společnosti Ignium, request_xml, která byla jako první nabízena jako prostředek pro komunikaci se serverem IPAS. Jelikož tato knihovna byla posouzena jako nedostačující a to především v oblasti aktualizace příkazů, vazeb a definic které je nutné dodržovat při komunikaci, vznikla knihovna clsIpas. clsIpas je distribuována ve zdrojovém kódu skrze svoji šablonu. Ta je základním kamenem pro dynamickou generaci clsIpas knihovny a příkazy pro komunikaci pomocí souboru pravidel rule.xml.

Knihovna clsIpas by měla sloužit především jako základní stavební kámen uživatelských rozhraní komunikujících se serverem IPAS.

.NET knihovna Ignium Kernel

Ignium Kernel je knihovna sloužící pro komunikaci se serverem IPAS z prostředí Windows, respektive z platformy Microsoft .NET. Knihovna je naprogramována v jazyce C#. Během vývoje byl kladen důraz na použitelnost i v jiných jazycích CLR. Je možné ji využít jak v dynamicky generovaných ASP.NET stránkách, tak desktopových aplikacích Windows Forms, konzolových aplikacích, příp. dalších oblastech, které .NET poskytuje.

Knihovna se skládá z objektů, které v sobě obsahují komunikační protokol serveru IPAS. Tyto objekty umožňují připojení k serveru, zaslání příkazů IPASu a přijímání odpovědí. Typické využití knihovny se tak skládá z vytvoření objektu pro připojení, vytvoření objektů s příkazy, voláním jedné metody a jejich odeslání serveru. Knihovnu lze využít i pro zaslání požadavku v jazyce XML, na kterém je protokol založen.

Pro své fungování knihovna potřebuje nainstalovaný .NET Framework.

Soubor RULE.XML

Soubor rule.xml obsahuje definice a pravidla pro komunikaci se serverem IPAS. Díky tomuto souboru je možné vytvořit validní kód příkazů a většinu možných chyb ošetřit již na straně klienta. Tím se šetří nutnost komunikace s IPASem.

O rozparsování souboru pravidel se stará generační skript, který je schopen z definovaných pravidel vytvořit jednotlivé příkazy s předem definovanými návratovými kódy a vstupním formátem dat. Ze souboru je rovněž možné získat formát příkazů i jiným způsobem.

Soubor sám obsahuje poznámky o typu pravidel. Jednoduché principy je možné snadno vysledovat např. z generačního skriptu knihovny clsIpas.

Komunikační protokol se serverem IPAS

Komunikačním protokolem serveru IPAS je jazyk XML. Veškerá komunikace probíhá v kódování UTF-8 podle definice XML 1.0.

Veškeré definice příkazů je možné získat v souboru *rule.xml*, který obsahuje jednotlivé GroupElementy a Elementy. Je třeba myslet na to, že jazyk XML je *CaseSensitive*.

Je třeba také myslet na nutnost překladu speciálních HTML znaků, které v jazyky xml nejsou povoleny. Nahrazení je třeba provést HTML ENTITAMI.

Knihovna clsIpas je připravena na to, přijímat data v předem definované kódové stránce a nepřeložené do htmlentities. Obojí pak provádí během přípravy XML příkazu, tak aby během komunikace nedošlo k porušení specifikace XML 1.0 a příkaz tak nebyl serverem odmítnut.

Hlavička komunikace

Každý request, tedy každé spojení se serverem je třeba zahájit hlavičkou. Tato hlavička je serverem kontrolována a je-li zjištěna jakákoliv chyba, je komunikace odmítnuta. Za hlavičkou již následují jednotlivé příkazy ve formátu XML. Hlavička obsahuje vždy 8 bytů a její formát je následující:

<i>Položka</i>	<i>Popis</i>
DWORD m_dwSize	Velikost požadavku bez hlavičky.
BYTE m_byOrder	Pořadí požadavku v otevřeném spojení.
BYTE m_byFormat	Formát požadavku (výchozí formát nebo gzip).
WORD m_wSum	Jednoduchý součet hlavičky.

Hlavička komunikace pro GZIP formát

Používáte-li GZIP spojení, je nutné před každý odesílaný souhrn příkazů přidat DWORD hlavičku obsahující velikost rozbaleného souhrnu dat. Úplná hlavička pro komprimovaná data tedy bude mít strukturu:

<i>Položka</i>	<i>Popis</i>
DWORD m_dwSize	Velikost požadavku bez hlavičky.
BYTE m_byOrder	Pořadí požadavku v otevřeném spojení.
BYTE m_byFormat	Formát požadavku (výchozí formát nebo gzip).
WORD m_wSum	Jednoduchý součet hlavičky.
DWORD m_dwFullSize	Velikost nekomprimovaného objemu dat.

Pravidla spojení

Během jednoho spojení k serveru IPAS je třeba dbát především na provedení příkazu Login. Tímto příkazem se autorizujete vůči serveru a autorizujete tak celé spojení až do jeho konce, příkazu LogOut nebo nového příkazu Login. V serveru IPAS2 je již role, ve které se uživatel přihlašuje řízena serverem a není ji tedy třeba předávat jako součást příkazu Login.

Testovací rozhraní

Pro účely ladění a testování je server IPAS2 nainstalován také v testovací instanci. Všechny objednávky zpracovávané testovací instancí serveru jsou buď vyřizovány přes testovací rozhraní správců jednotlivých TLD nebo jsou pouze přijaty, ale nejsou nikdy vyřízeny.

Testovací server je dostupný na IP adrese 217.31.49.19 a portu 5155 (produkční server je dostupný na adrese 217.31.49.19, portu 5055).

Zřízení testovacího účtu

Na adrese <http://www.domena.cz/public/ipas2/index.php> si můžete zřídit uživatelský účet pro přístup k testovacímu serveru. Tento účet slouží pouze pro přístup k testovacímu serveru a má nastaven počáteční kredit ve výši 20000,-Kč. Tento kredit je možné použít pro uhrazení testovacích objednávek, aby tyto mohly být zpracovány.

Dostupná funkcionality

Následující tabulka uvádí přehled dostupných funkcí v testovacím systému:

<i>Funkcionalita</i>	<i>Popis</i>
Registrace EU kontaktů a domén	K dispozici je plně funkční testovací rozhraní EURid.
Registrace CZ kontaktů, sad nameserverů a domén	K dispozici je plně funkční testovací rozhraní CZ.NIC.
Registrace generických kontaktů a domén	Funkční je pouze přijetí objednávky serverem IPAS2, vyřízení není možné z důvodu absence testovacího rozhraní správce TLD.
Registrace virtuálních kontaktů	K dispozici je plně funkční testovací rozhraní.
Registrace ostatních domén	Funkční je pouze přijetí objednávky serverem IPAS2, vyřízení není možné.

Testování na produkčním systému

Rozhraní je možné testovat také přes účet, který máte vytvořen na systému <http://www.domena.cz>. Je třeba si však uvědomit, že takové objednávky se následně skutečně vyřizují přes ostrý systém a tudíž jednotlivé platby jsou strhávány z Vašeho kreditu. Výhodou tohoto testování je možnost kontrolovat si objednávky přes <http://www.domena.cz>.

Popis PHP knihovny cslspas

Jelikož je knihovna napsána v jazyce PHP 5, která již podporuje zapouzdření proměnných či metod tříd, je možné využít jenom některé proměnné nebo metody. Tuto skutečnost snadno zjistíte z přiložených příkladů komunikace.

Přehled nejdůležitějších změn pro IPAS2

- Během přihlašování uživatele již není třeba uvádět jeho roli v systému.
- Registrace kontaktů, domén, sad nameserverů, jejich změny a údržba probíhá přes volání serveru OrderService.
- Změna způsobu pro odblokování účtu při ztrátě hesla – nedochází k zaslání původního hesla uživateli, uživatel si musí heslo změnit.

Popis jednotlivých tříd knihovny cslspas

Popis třídy cslspas

Instanční proměnné třídy

<i>Proměnné socket spojení</i>	
\$hCoreSocket = False	Ukazatel připojení k jádru.
<i>Proměnné ovládající GZIP chování</i>	
\$bGZData = false	Definuje defaultní chování GZIP komprese.
\$GZLevel = 9	Určuje úroveň komprese (1 – minimální komprese, 9 – maximální komprese).
\$iGZMoreThan = 1024	Komprimuje pouze příkazy větší než zadaná hodnota (v bytech bez hlavičky).
\$aGZAllways = Array('Login')	Definuje pole příkazů, které se zagzipují vždy, nezávisle na hodnotě v \$bGZData.
\$bGZDataNow = false	Slouží k ukládání stavu komprese pro aktuálně odesílaný request.
<i>Proměnné určující místo a název souborů příkazů</i>	
\$sRequestFcePrefix = '<%sRequestFcePrefix%>'	Předpona funkcí jednotlivých requestů (slouží k zabránění konfliktů s funkcemi Vašeho vlastního systému) .
\$sRequestFileExtend = '<%sPostFixRequestTmp%>'	Přípona souborů obsahujících příkazy a jejich definice.
\$mExternalFunctionsPath = "	Proměnná s relativním adresou k umístění souborů s definicí příkazů.
<i>Počítadla příkazů</i>	
\$iRequestCounter = 0	Počítadlo příkazů v požadavku.
\$iRequestSendCounter = 0	Počítadlo příkazů v otevřeném spojení.
<i>Proměnné pro přípravu příkazů</i>	
\$aRequestData = Array()	Proměnná pro vstupní data požadavků.
\$aRequestOut = Array()	Proměnná s výsledným XML požadavkem.
\$sRequestCache = "	Cache těla příkazu.
<i>Proměnné pro příchozí data</i>	
\$sRequestResult = "	Výsledná XML odpověď z jádra.
<i>Proměnné XML data a index</i>	
\$aParseXmlValues = Array()	Pole hodnot výstupu XML parseru.

\$aParseXmlIndexes = Array()	Pole indexů XML parterů.
Data pro znakovou konverzi	
\$aCharsetBoth = Array()	Pomocné pole pro překlad kódových stránek.
\$aCharsetIn = Array(<%sCharsetIn%>)	Pole s problematickými znaky ve vnitřní kódové stránce.
\$aCharsetOut = Array(<%sCharsetOut%>)	Pole s problematickými znaky v UTF-8.

Metody třídy

<i>Metoda</i>	<i>Popis</i>
<code>__construct()</code>	Konstruktor třídy.
<code>__destruct()</code>	Destruktor třídy.
<code>strCreateRequest()</code>	Vytvoří z aktuálních dat v <i>\$aRequestData</i> pole <i>\$aRequestOut</i> . Nedošlo-li k chybě, návratem bude string se XML requestem. V případě chyby bude na výstupu pole s chybovými kódy jednotlivých elementů příkazu.
<code>UnCallExternalFunction(\$sRequestName, array \$aInRequestData = "")</code>	Vyhledá a zavolá funkci s definicí daného příkazu. Není-li nalezen příkaz, nebo dojde-li k chybě, vrací false. Jinak vrací návratovou hodnotu volané funkce.
<code>arrayConvertInData(&\$aInData)</code>	Je rekurzivním překladačem vstupních dat na správné htmlentities.
<code>strConvertData(\$sInData, \$sOrder)</code>	Konvertuje data mezi kódovými stránkami za pomoci definovaných polí problematických znaků.
<code>strConvertRequest()</code>	Skládá data z proměnné <i>\$aRequestOut</i> do výsledného XML stringu.
<code>voidParseXml()</code>	Plní datové a indexové pole rozparsovaným XML odpovědi jádra.
<code>strCreateHeader()</code>	Vytváří hlavičku požadavku.
<code>Command()</code>	Odešle požadavek na server a zpracuje odpověď.
<code>OpenConnect()</code>	Otevře spojení k serveru.
<code>CloseConnect()</code>	Uzavře spojení k serveru
<code>ResetSession()</code>	Resetuje aktuální spojení, případně vytvoří nové.

Popis třídy *IpasElementFunction*

Třída *IpasElementFunction* slouží k možnosti nastavení daného elementu requestu jako vnitřní funkce *Ipasu*. Vnitřní funkce *Ipasu* slouží především k překladu hodnot typu *DomainName* či *RRID* na hodnoty jako *ID*, které tak zjednodušují práci programátora a šetří užití příkazu *Query*.

Funkčnost

Zadáte-li jako hodnotu elementu objekt typu *IpasElementFunction*, element se zpracuje poněkud odlišným způsobem. Takový element bude vytvořen s atributem `function="1"` a hodnotou představující klasické volání dané funkce. *Ipas* sám pak nejprve provede požadavek na volání funkce, její výstup dosadí za daný element a pak teprve zpracovává celý příkaz již klasickým způsobem.

Implementace

Třída *IpasElementFunction* obsahuje seznam funkcí, které jsou v *Ipasu* momentálně implementovány. Zároveň obsahuje seznam a typ atributů, které daná funkce vyžaduje. Typy atributů jsou rozděleny pouze jako *Integer (I)* a *String (S)*, které se při generování volání funkce (a tedy obsahu daného elementu) liší užitím uvozovek. Vytvoříte-li objekt tohoto typu s uvedením chybného názvu funkce, bude při generování jeho výstupní hodnoty (tedy při užití v *Requestu*) vyvolána výjimka *IpasException*.

Příklad

Následující příklad ukazuje použití funkce IPAS ze třídy *IpasElementFunction* pro získání ID dotazu do databáze na základě znalosti jeho identifikátoru.

Vzorový program

```

$Ipas = new clsIpas();
$queryID = new IpasElementFunction('QueryCodeToID', array('Code' => '45'));
$reqData[0]['name'] = 'Query';
$reqData[0]['data'] = array(
    'ID' => $queryID,
    'Params' => array('ORDER' => 'date')
);
$Ipas->aRequestData = $reqData;
$request = $Ipas->strCreateRequest();
print_r($request);

```

Vygenerovaný XML příkaz

```

<?xml version="1.0" encoding="UTF-8" ?>
<Commands>
  <Command ID="0">
    <Query>
      <ID function="1">QueryCodeToID("48")</ID>
      <Params>
        <ORDER>date</ORDER>
      </Params>
    </Query>
  </Command>
</Commands>

```

Popis přidanych funkcí ve třídě *IpasAddFunction*

Metoda	Popis
arrMultiArrayToSingleArray	Slouží k přegenerování vícerozměrného pole do jednorozměrného. Více rozměrů je interpretováno pak interpretováno tečkou v klíči daného pole. Vstupem je pole předávané hodnotou. Výstupem je pak pole upravené.
arrQuery	Slouží k volání příkazu <i>Query</i> pomocí třídy <i>clsIpas</i> . Výstupem je jednoduché a již rozparované pole hodnot, které <i>Ipas</i> vrátí v XML. Vstupem je instance komunikační třídy <i>clsIpas</i> , skrze kterou se příkaz provede, kód <i>Query</i> příkazu a pole dalších parametrů pro dosažení do <i>Query</i> . V této funkci je možné vidět použití <i>IpasElementFunction</i> , které zde slouží k překladu <i>QueryCode</i> na <i>QueryID</i> .
arrFromXML	Je funkcí umožňující přegenerování dvou polí, vznikajících po rozparování XML skrze vnitřní parser PHP, na více rozměrné pole. Vstupem je indexové a hodnotové pole vzniklého po parsování (v <i>clsIpas</i> je to <i>aParseXmlIndexes</i> a <i>aParseXmlValues</i>) a název tagu, od kterého se má výstup provádět.

<i>Metoda</i>	<i>Popis</i>
	Název tagu slouží k vymezení „zanoření“ do XML. Výstupem je vícerozměrné pole. Tato funkce ignoruje hodnoty atributů jednotlivých elementů Xml výstupu. Atributy jsou využity v podstatě pouze u příkazu <i>CheckDomain</i> .
arrIpasFromXML	Tato funkce je pouze aliasem k funkci <i>arrFromXML</i> . Její vstupní hodnotou však nejsou pole indexů a hodnot, ale instance třídy <i>clsIpas</i> . Funkce sama pak pouze volá <i>arrFromXML</i> s uvedenými poli z dodaného objektu. Výstup je totožný.

Dynamické generování příkazů a knihovny *clsIpas*

Dynamické generování příkazů knihovny *clsIpas* je založena zejména na souboru pravidel *rule.xml*. Ten obsahuje definice jednotlivých příkazů podporovaných IPAS serverem, včetně validačních pravidel pro jejich volání. To nám umožňuje snadno přidávat nově vyžadovanou funkcionalitu a zároveň tak omezit množství vznikajících chyb a dalších dotazů.

Součástí knihovny *clsIpas* je proto také skript *generate.php*, který na základě souboru *rule.xml* vygeneruje třídy a funkce v jazyce PHP potřebné pro volání jednotlivých funkcí serveru IPAS. Chování generačního skriptu je možné ovlivnit nastavením v konfiguračním souboru *generate.var*.

Konfigurace generátoru knihovny

Konfigurace generování knihovny se provádí pomocí souboru *generate.var*, jak již bylo zmíněno výše. Význam jednotlivých proměnných je uveden zde:

<i>Název proměnné a výchozí hodnota</i>	<i>Popis</i>
<code>\$\$FileRule = './rule.xml'</code>	Soubor s pravidly podle kterých se generují jednotlivé příkazy.
<code>\$\$FileClass = './clsIpas.tpl'</code>	Soubor s šablonou knihovny <i>clsIpas</i> .
<code>\$\$NameTestClass = 'clsTest'</code>	Název pro třídu s jednoduchými testy parametrů příkazů.
<code>\$\$NameCommClass = 'clsIpas'</code>	Název pro třídu komunikace se serverem <i>Ipas</i> .
<code>\$\$PostFixClass = 'class'</code>	Přípona souborů se třídami (<i>clsIpas</i> a <i>clsTest</i>).
<code>\$\$PostFixRequest = 'class'</code>	Přípona souborů s jednotlivými příkazy.
<code>\$\$RequestFcePrefix = 'IpasReq'</code>	Předpona pro název funkcí jednotlivých příkazů.
<code>\$\$RequestDir = 'request'</code>	Umístění souborů s příkazy vůči knihovně.
<code>\$\$DataDir = 'data'</code>	Umístění vygenerovaných souborů vůči generačnímu skriptu.
<code>\$\$bGenerateRequest = true</code>	Přepínač určující zda má skript generovat soubory s příkazy.
<code>\$\$bGenerateCommClass = true</code>	Přepínač určující zda má skript generovat třídu <i>clsIpas</i> .
<code>\$\$IpasServer = 'offline.ignum.cz'</code>	Adresa IPAS serveru.
<code>\$\$IpasPort = '5055'</code>	Port serveru IPAS
<code>\$\$WebCharset = 'WINDOWS-1250'</code>	Kódová stránka uživatelského rozhraní. Vzhledem k tomu, že překlad se provádí pomocí funkce <i>iconv</i> , je třeba názvy kódových stránek volit z možných nastavení této funkce. Upozorňujeme, že vzhledem ke způsobu překladu není v tuto chvíli možné zvolit kódovou stránku UTF-8. Tento nedostatek odstraníme v některé z dalších verzí šablony <i>clsIpas</i>.
<code>\$\$ConnectionType = 'norm'</code>	Instance serveru IPAS2, která bude používána. Defaultně nastaveno na

Název proměnné a výchozí hodnota	Popis
	produkční systém. Pokud je hodnota <i>test</i> , používá se testovací instance IPAS2 serveru.

Princip tvorby knihovny a příkazů

Pro Vaši snazší orientaci, je jistě dobré znát něco malinko z principu generování knihovny a příkazů.

Zatímco třída *clsIpas* se prakticky negeneruje (pouze se vyplní některé z hodnot - ty rozpoznáte snadno podle oddělovačů `<% a %>`), soubory s příkazy se generují zcela kompletně. Navíc, vzhledem k tomu, že dochází k prolínání se některých příkazů či spíše group elementů do více příkazů je nutné sledovat tyto vazby. Program tedy prochází každé pravidlo, zjišťuje jeho typ, validační hodnoty a vytváří podle toho PHP kód dané funkce. Navíc, chce-li využít jiné funkce, které nejsou umístěné v daném souboru, vytvoří mezi oběma soubory jednosměrnou vazbu. Při volání daného příkazu je pak soubor s příkazem requireován čímž se zadefinuje funkce složená z předpony definované v *generate.var* a názvu příkazu. Tento princip pak zjednodušuje a zrychluje použití knihovny především při volání pouze několika příkazů či volání mnoha stejných příkazů.

Z našich pokusů vyplynulo, že rozdělení do více souborů znamená při běžném využití knihovny pro připojení k serveru úsporu více než dvojnásobnou, oproti umístění všech příkazů do jednoho souboru (potažmo knihovny). To je způsobeno výrazným ušetřením množství kódu, který je třeba systémem opakovaně kompilovat.

Zároveň s těmito soubory vzniká soubor defaultně pojmenovaný *clsTest.class*, který obsahuje knihovnu sdružující nejelementárnější validační pravidla definované v souboru pravidel *rule.xml*.

Instalace knihovny

1. Stáhněte si aktuální verzi knihovny z adresy <ftp://ftp.domena.cz/clsIpas/latest/clsIpas.zip> a uložte si ji na disk.
2. Rozbalte knihovnu do zvoleného adresáře na disku (např. `C:\clsIpas`).
3. V případě potřeby proveďte změny v konfiguračním souboru *generate.var*.
4. Spusťte si Příkazový řádek (Nabídka Start -> Programy -> Příslušenství -> Příkazový řádek).
5. Zadejte příkaz (v případě, že je knihovna v jiném adresáři, pak nahraďte cestu): `cd c:\clsIpas`
6. Nyní spusťte příkaz (pokud máte PHP5 instalováno v jiné složce, pak opět nahraďte cestu): `c:\php5\php.exe generate.php`
7. V podsložce *data* by nyní měly být vygenerovány všechny potřebné třídy a příkazy knihovny. Tato podsložka v podstatě obsahuje funkční kopii knihovny *clsIpas* určenou k připojení k serveru IPAS.

Volání příkazů serveru IPAS

Volání příkazů serveru IPAS lze rozdělit na dvě části:

1. vygenerování XML požadavku na základě vstupních parametrů a kontrola výstupu generátoru (odhalí případné chybějící/špatně vyplněné údaje).
2. volání příkazu serveru IPAS a kontrola návratových hodnot.

Návratové hodnoty generátoru XML příkazů pro server IPAS

Při komunikaci se serverem pomocí generovaných příkazů, dochází k validaci již na straně klienta (během generování XML požadavku pro server), což výrazně snižuje čas potřebný pro zpracování požadavku a zároveň snižuje požadavky na připojení k serveru IPAS. Skripty vrací chybové kódy vždy ve vztahu k elementu jehož se chyba týká. Jedná-li se o vnořený element, bude chyba také ve složeném poli.

Hlavní snahou je vždy vrátit maximum chybových kódů, tedy projít všechny validační mechanismy a sestavit z nich

případně úplný seznam chyb ve vstupních datech. Kódy jsou vráceny v poli, jehož klíč je shodný s názvem elementu a hodnoty jsou (nejedná-li se o již zmiňovaný vnořený element) umístěny ve dvou klíčích. První a vždy povinný je klíč code. Druhý nepovinný je z výběru klíčů format, value.

Vzhledem k množství jednoduchých validačních metod budou tyto v dalších verzích dále implementovány a budou tedy vznikat i další chybové kódy.

Chybové kódy jsou v tuto chvíli rozděleny tak aby odpovídaly programové pozici a typu validace. Kódy 2xx jsou z nejjednodušších validačních metod, kód 3xx jsou ze složených validačních metod a kódy 4xx jsou používány pro chyby elementů. Následující tabulka uvádí základní přehled chybových kódů uložených v klíči code:

<i>Hodnota</i>	<i>Popis</i>
201	Hodnota není integer (momentálně se tato kontrola nepoužívá).
202	Hodnota není z hodnot 'True', 'False' (string boolean).
300	Hodnota je příliš krátká (klíč format obsahuje minimální délku).
301	Hodnota je příliš dlouhá (klíč format obsahuje maximální délku).
302	Hodnota není z možných hodnot (klíč format obsahuje možné hodnoty oddělené čárkou).
303	Hodnota je příliš malá (klíč format obsahuje nejmenší možnou hodnotu).
304	Hodnota je příliš dlouhá (klíč format obsahuje největší možnou hodnotu).
305	Hodnota neodpovídá předepsanému datovému formátu (klíč format obsahuje předepsaný datový formát).
306	Hodnota neodpovídá předepsanému regulárnímu výrazu (klíč format obsahuje předepsaný regulární výraz).
400	Prázdná povinná hodnota.
401	Neznámá hodnota neodpovídající žádné z možných hodnot přepínače (klíč value obsahuje chybový kód serveru Ipas, který by v takovém případě vrátil).
402	Data tohoto elementu musí být pole.
403	Element je v nedostatečném množství (klíč format obsahuje minimální množství elementu).
404	Element je v příliš vysokém množství (klíč format obsahuje maximální množství elementu).

Seznam příkazů

Server Ipas momentálně podporuje tyto příkazy přístupné pro partnery nebo subregistrátory.

<i>Příkaz</i>	<i>Popis</i>
AcceptAccountAccess	Potvrzení spojení accountů (domena.cz – správa plátců)
AssignContact	Import kontaktu
AssignDomain	Import domény
AssignSubject	ZRUŠENO
AssignVirtualContact	Import virtuálního kontaktu
AutoRenewDomain	Zapnutí/vypnutí automatické fakturace domény při expiraci
CancelAccountAccess	Zrušené spojení accountů (domena.cz – správa plátců)
CancelInvoice	Storno výzvy k platbě

<i>Příkaz</i>	<i>Popis</i>
CancelOrder	Storno objednávky
ChangeAccountPassword	Změna hesla
CheckContact	Zjištění existence kontaktu
CheckDomain	Zjištění existence domény
CheckSubject	ZRUŠENO
ConfirmOrder	Potvrzení objednávky
CreateAccount	Vytvoření accountu (uživatelského účtu)
CreateContact	ZRUŠENO (nahrazuje jej příkaz OrderService)
CreateDomain	ZRUŠENO (nahrazuje jej příkaz OrderService)
CreateFinalInvoice	Vytvoření daňového dokladu
CreateInvoice	Vytvoření výzvy k platbě
CreateNicAgreement	ZRUŠENO
CreateNServer	Vytvoření DNS serveru
CreateSubject	ZRUŠENO
CreateVirtualContact	Vytvoření virtuálního kontaktu
CreditNoteGetImage	Zobrazení dobropisu
DeleteContact	ZRUŠENO
DeleteDomain	Zrušení (úplné) domény ZRUŠENO (nahrazuje jej příkaz OrderService)
DeleteNServer	Smazání DNS serveru
DeleteSubject	ZRUŠENO
DeleteVirtualContact	Smazání virtuálního kontaktu
DetachContact	Odstranění kontaktu ze seznamu daného účtu
DetachDomain	Odstranění domény ze seznamu daného účtu
DetachSubject	ZRUŠENO
DetachVirtualContact	Odstranění virtuálního kontaktu ze seznamu daného účtu
GetAccountInfo	Zobrazit informace o accountu
GetContactInfo	Zobrazit informace o kontaktu podle RRID
GetContactInfoByID	Zobrazit informace o kontaktu podle ID
GetDomainInfo	Zobrazit informace o doméně podle názvu domény
GetDomainInfoByID	Zobrazit informace o doméně podle ID
GetNicAgreementVersions	ZRUŠENO
GetSubjectInfo	ZRUŠENO
GetSubjectInfoByID	ZRUŠENO
GetVirtualContactInfo	Zobrazit informace o virtuálním kontaktu
InvoiceGetImage	Zobrazit výzvu k platbě
InvoiceTaxGetImage	Zobrazit daňový doklad

<i>Příkaz</i>	<i>Popis</i>
JoinAccount	Sloučit účet
Login	Přihlášení se
Logout	Odhlášení se
MakeInvoicePaid	Zaplatit výzvu k platbě
OrderManualFinish	Potvrzení manuální objednávky
OrderManualStart	Označení začátku zpracování manuální objednávky
OrderService	Objednání služby
Query	Spuštění SELECTu
RenewDomain	Prodloužení domény ZRUŠENO (nahrazuje jej příkaz OrderService)
RepairRRID	ZRUŠENO
RequestAccountAccess	Požadavek na spojení accountů (domena.cz – správa plátců)
ResendConfirm	Přeposlání potvrzení objednávky
ResendInvoice	Přeposlání výzvy k platbě/daňového dokladu
SendLoginInfo	Přeposlání přihlašovacích údajů
SetCurrentAccount	Přihlášení se pod Account
TransferDomain	Transfer domény ZRUŠENO (nahrazuje jej příkaz OrderService)
UpdateAccount	Upravit údaje accountu
UpdateContact	ZRUŠENO (nahrazuje jej příkaz OrderService)
UpdateDomain	Úprava domény ZRUŠENO (nahrazuje jej příkaz OrderService)
UpdateNServer	Úprava DNS serveru
UpdateSubject	ZRUŠENO
UpdateVirtualContact	Úprava virtuálního kontaktu
ViewLogin	Zobrazit informace o přihlášeném účtu
GetNSSetInfo	Zobrazit informace o NSSETu se zadaným RRID
GetNSSetInfoByID	Zobrazit informace o NSSETu se zadaným interním ID IPASu

Soubor *rule.xml* obsahuje i další příkazy, které však není bez oprávnění Admin možné volat. Při pokusu o zavolání takového příkazu server odmítne tento příkaz vykonat.

Podrobný popis důležitých příkazů

Příkaz Login

<i>Parametr</i>	<i>Povinný</i>	<i>Popis</i>
LoginName	Ano	Přihlašovací jméno účtu
Password	Ano	Heslo účtu

Příkaz GetDomainInfo

Parametr	Povinný	Popis
Domain	Ano	Název domény jejíž informace chcete zobrazit.

Příkaz CheckDomain

Parametr	Povinný	Popis
Name	Ano	Je MultiElementem, tedy elementem, který se může libovolně opakovat. Minimálně se musí vyskytnout právě jednou. Obsahuje název domény, o které chcete vědět, zda je volná či nikoliv. Jelikož jde o MultiElement, musí být vstupní data do tohoto elementu polem.

Příkaz Query

Parametr	Povinný	Popis
ID	Ano	Určuje SELECT, který chcete spustit. Seznam základních SELECTů je uveden mezi přílohami. Další SELECTy je možné získat na naší technické podpoře.
Database	Ne	Je na výběr z hodnot <i>data</i> a <i>log</i> . Hodnotou <i>data</i> určujete datovou databázi, hodnotou <i>log</i> databázi logů. Defaultní hodnotou je <i>data</i> .
Params	Ne	Je GroupElementem, který slučuje všechny klíče potřebné ke spuštění SELECTu. Tyto klíče zjistíte také na naší technické podpoře, společně s jednotlivými SELECTy. Klíče uvnitř tohoto elementu musí být názvem shodné s klíčem v SELECTu. Některé klíče systém předdefinovává automaticky (např. <i>AccID</i> – ID přihlášeného uživatele, <i>AccLogin</i> – obsahuje login aktuálního uživatele, apod.). Tyto parametry jsou pak systémem přepsány a na přijaté parametry tedy není brán zřetel.

Příklady

Přihlášení k serveru IPAS

```
$oMyIpas = new clsIpas();
$aReqData[0]['name'] = 'Login';
$aReqData[0]['data'] = Array(
    'LoginName' => 'slon',
    'Password' => 'bufamasvousama'
);
$oMyIpas->aRequestData = $aReqData;
$aOutReq = $oMyIpas->strCreateRequest();
If (is_array($aOutReq)) { // kontrola vygenerovaného XML příkazu -> pokud
    // pole - chyba vstupních parametrů
    strZpracujChyby($aOutReq); //fiktivni funkce pro zpracování chyb
} else {
    $oMyIpas->Command();
}
```

Volání příkazu IPAS GetDomainInfo

```
<?
require_once 'clsIpas.class';

$oMyIpas = new clsIpas(); // nejprve si zajistím přihlášení
$aReqData[0]['name'] = 'Login';
$aReqData[0]['data'] = Array(
```

```
                'LoginName' => 'slon',
                'Password' => 'bufamasvousama'
            );
$omyipasa->aRequestData = $areqdata;
$aoutreq = $omyipasa->strCreateRequest();

$omyipasa->Command();
$aReqData[0]['name'] = 'GetDomainInfo';
$aReqData[0]['data'] = array('Domain' => 'domena.cz');
$omyipasa->aRequestData = $areqdata;
$aoutreq = $omyipasa->strCreateRequest();
if (is_array($aoutreq)) { // kontrola vygene XML příkazu -> pokud pole, chyba vstupu
    print_r($aoutreq);
    die;
} else
    $omyipasa->Command();

$iaf = new IpasAddFunction(); // konverze odpovědi od IPASu na PHP pole
print_r($iaf->arrIpasFromXML($omyipasa, 'DomainInfo')); // zobrazím si odpověď v poli
?>
```

Volání příkazu IPAS CheckDomain

```
<?
header('Content-type: text/plain');
require_once 'clsIpas.class';

$omyipasa = new clsIpas(); // nejprve si zajistím přihlášení
$aReqData[0]['name'] = 'Login';
$aReqData[0]['data'] = Array(
    'LoginName' => 'slon',
    'Password' => 'bufamasvousama'
);
$omyipasa->aRequestData = $areqdata;
$aoutreq = $omyipasa->strCreateRequest();
$omyipasa->Command();
$aReqData[0]['name'] = 'CheckDomain';
$aReqData[0]['data'] = array('Name' => array('domena.cz', 'nejakaprazdnadomena.eu'));
$omyipasa->aRequestData = $areqdata;
$aoutreq = $omyipasa->strCreateRequest();
if (is_array($aoutreq)) { // kontrola vygene XML příkazu -> pokud pole, chyba vstupu
    print_r($aoutreq);
    die;
} else
    $omyipasa->Command();
print_r ($omyipasa->sRequestResult);
?>
```

Volání příkazu IPAS Query

```
<?
require_once 'clsIpas.class';

$omyipasa = new clsIpas(); // nejprve si zajistím přihlášení
$aReqData[0]['name'] = 'Login';
$aReqData[0]['data'] = Array(
    'LoginName' => 'slon',
    'Password' => 'bufamasvousama'
);
$omyipasa->aRequestData = $areqdata;
$aoutreq = $omyipasa->strCreateRequest();
```

```
$oMyIpas->Command();
$aReqData[0]['name'] = 'Query';
$aReqData[0]['data']['ID'] = new IpasElementFunction('QueryCodeToID', array('Code' => '350'));
$aReqData[0]['data']['Params'] = array('AccID' => 48);
$oMyIpas->aRequestData = $aReqData;
$aOutReq = $oMyIpas->strCreateRequest();

if (is_array($aOutReq)) { // kontrola vygener XML příkazu -> pokud pole, chyba vstupu
    print_r($aOutReq);
    die;
} else
    $oMyIpas->Command();

$iaf = new IpasAddFunction(); // konverze odpovědi od IPASu na PHP pole
$rows = $iaf->arrIpasFromXML($oMyIpas, 'Row');
// vypisu jednotlivé domeny a datum konce platnosti
foreach($rows as $row)
    echo $row['name'] . " " . $row['valid_to'] . "<br />\n";
?>
```

Volání příkazu OrderService pro vytvoření generického kontaktu

```
<?
require_once 'clsIpas.class';

$oMyIpas = new clsIpas(); // nejprve si zajistím přihlášení
$aReqData[0]['name'] = 'Login';
$aReqData[0]['data'] = Array(
    'LoginName' => 'slon',
    'Password' => 'bufamasvousama'
);
$oMyIpas->aRequestData = $aReqData;
$aOutReq = $oMyIpas->strCreateRequest();
$oMyIpas->Command();
$aReqData[0]['name'] = 'OrderService';
$aReqData[0]['data']['ServiceCode'] = 'Contact.JOKER.Create';
$aReqData[0]['data']['TldID'] = 2; // 2 - .com, 3 - .net, 4 - .org, 5 - .biz, 6 - .info
$aReqData[0]['data']['Organization'] = 'My Company';
$aReqData[0]['data']['FirstName'] = 'Jmeno';
$aReqData[0]['data']['LastName'] = 'Prijmeni';
$aReqData[0]['data']['Individual'] = 'False'; // True/False
$aReqData[0]['data']['EMail'] = 'my@company.cz';
$aReqData[0]['data']['Phone'] = '+420.123123123';

$adr = array(
    'Name' => 'My Company',
    'Street' => 'Prague street 2',
    'City' => 'Prague',
    'ZIP' => '11150',
    'CountryID' => 203 // Czech Republic
);

$aReqData[0]['data']['CreateAddress'] = $adr;

$oMyIpas->aRequestData = $aReqData;
$aOutReq = $oMyIpas->strCreateRequest();

if (is_array($aOutReq)) { // kontrola vygener XML příkazu -> pokud pole, chyba vstupu
    print_r($aOutReq);
    die;
}
```

```
} else
    $oMyIpas->Command();

echo "Ipas response: " . $oMyIpas->sRequestResult;
?>
```

Volání příkazu OrderService pro registraci generické domény

```
<?
require_once 'clsIpas.class';

$oMyIpas = new clsIpas(); // nejprve si zajistím přihlášení
$aReqData[0]['name'] = 'Login';
$aReqData[0]['data'] = Array(
    'LoginName' => 'slon',
    'Password' => 'bufama'
);
$oMyIpas->aRequestData = $aReqData;
$aOutReq = $oMyIpas->strCreateRequest();

$oMyIpas->Command();

$aReqData[0]['name'] = 'OrderService';
$aReqData[0]['data'] = array (
    'ServiceCode' => 'Domain.COM.Create.1', // COM domena na 1 rok (max. 10)
    'Name' => 'mynewdomain234.com',
    'OwnerID' => new IpasElementFunction('ContactRRIDToID', array('ContactRRID' => 'C460122-
LRMS', 'TldID' => 2)), // RR_ID kontaktu vlastnika nebo lze pouzit primo ID
    // 'AdminID' => ... nepovinne
    // 'TechID' => ... nepovinne
    'AttachNServer' => array( // alespon 2x NS - seznam lze zjistit napr. pomoci Query 28
        array('ID' => 1, 'Type' => 'P'), // primarni NS
        array('ID' => 2, 'Type' => 'S'), // sekundarni NS
    ),
);

$oMyIpas->aRequestData = $aReqData;
$aOutReq = $oMyIpas->strCreateRequest();

if (is_array($aOutReq)) { // kontrola vygener XML příkazu -> pokud pole, chyba vstupu
    print_r($aOutReq);
    die;
} else
    $oMyIpas->Command();

echo "Ipas response: " . ($oMyIpas->sRequestResult);

?>
```

Přehled změn v knihovně

Tabulka uvádí přehled starších úprav knihoven. Aktuální informace o změnách jsou dostupné v souboru Changelog.txt distribuovaném s knihovnou.

<i>Verze</i>	<i>Změna</i>	<i>Detail</i>
2.3.0 (20061023)	Změna 'generate.php' Změna 'rule.xml'	Opraveno zpracování souboru rule.xml pro IPAS2 Změna definic příkazů a služeb pro IPAS2. Obsahuje velké množství změn v jednotlivých příkazech a parametrech, které musejí být předány při volání.

<i>Verze</i>	<i>Změna</i>	<i>Detail</i>
2.2.1 (20051012)	Třída Query 'clsIpas.tpl'	Opravena chyba v Query zajišťující překlad CodeToID.
2.2.0 (20050727)	Změna 'generate.php'	Změněn způsob generování překladových polí.
2.1.0 (20050722)	Změna 'clsIpas.tpl'	Opravena chyba znemožňující správné rozparsování českých znaků XML Parserem. Přidáno ověření existence spojení před jeho zavřením. Přidána funkce pro rekurzivní překlad polí.
2.0.1 (20050712)	Změna 'clsIpas.tpl'	Opravena chyba znemožňující správné vyvolání Exception (použití @).
2.0.0 (20050623)	Změna 'clsIpas.tpl'	Upravena pro funkčnost v PHP5 (>=5.0.2). Přidána podpora funkcí (třída IpasElementFunction). Přidána třída s podpůrnými funkcnostmi (parsing XML, Query) (třída IpasAddFunction). Možnost spojení SSL. Ošetření chyb skrze Exception a IpasException.
	Změna 'generate.php'	Přidána práce s funkcemi jako daty Elementu. Přidána volba typu spojení (generate.var). Přesunutí ChangeLogu.
1.1.2 (20050510)	Oprava 'generate.php'	Oprava chyby volání Inherit Typu Elementu.
1.1.1 (20050415)	Oprava 'generate.php'	Oprava tvorby requestu OrderService (překlad ServiceCode a class_id).
1.1.0 (20050331)	Oprava 'generate.php'	Změna způsobu jazykové konverze. Oprava 'quantitymax'.
1.0.3 (20041026)	Oprava 'generate.php'	Opraven chybějící ')' a ';' ve vygenerovaných funkcích requestů.
1.0.2 (20041025)	Oprava	Opravena chyba v názvu proměnné ve funkci strConvertData.
1.0.1 (20040924)	IPAS (20040927)	Opravena chyba zabraňující plnohodnotné komunikaci v UniCode.
	Přidáno	Schopnost komunikovat s GZIP komprimovanými příkazy. Hodnoty \$bGZData, \$iGZLevel, \$iGZMoreThan, \$aGZAllways ovládající GZIPování a GZIPovaný výstup. Hodnota \$\$RequestCache zrychlující generování příkazů (použito ve funkcích Command, strCreateRequest, strConvertRequest).
	Změny	Volání trigger_error ve výjimečných chybových situacích. Příkaz vrácený funkcí strCreateRequest je vrácen ve skutečné formě. Při použití GZIPu je tedy zazipovaný (pak obsahuje i DWORD hlavičku). Funkce Command automaticky detekuje typ dat vrácených serverem a zazipovaná data automaticky rozbálí. Kosmetické přejmenování funkce voidCallExternalFunction na unCallExternalFunction odpovídající typu vrácených dat.

<i>Verze</i>	<i>Změna</i>	<i>Detail</i>
	Oprava	Funkce strConvertData nově používá funkce strtr metodu pro překlad dat (nahrazuje funkci replace str_replace v jednom směru a For cyklus v druhém směru překladu). Opravena chyba v překlepu ve funkci strConvertData (chyba v přípravě pole pro překlad dat).
0.9.1 (20040915)	První veřejně dostupná verze	Základní funkcionality pro komunikaci, zpracování příkazů.

Popis .NET knihovny IGNUM KERNEL

Knihovna Ignum Kernel se skládá z objektů, které zapouzdřují generování requestů pro IPAS server. V případě, že během volání dojde k chybě může dojít k vyvolání výjimky. Pro připojení a komunikaci se serverem slouží třída `Ignum.Kernel.DirectSession`. Po připojení k serveru je třeba navázat spojení pomocí metody `Start()`. V případě, že příkaz slouží ke zjišťování informací (např. `GetDomainInfo`), pak jsou požadované informace obvykle po provedení příkazu dostupné ve formě `properties` objektu příkazu.

Popis třídy `Ignum.Kernel.DirectSession`

Třída slouží k navázání spojení se serverem IPAS a umožňuje volání jednotlivých příkazů serveru. Význam nejdůležitějších metod je v následující tabulce:

<i>Metoda</i>	<i>Argument</i>	<i>Popis</i>
<code>DirectSession(string host, int port)</code>		Konstruktor s nastavením připojovacích informací k serveru IPAS.
	<code>host</code>	Adresa serveru IPAS.
	<code>port</code>	TCP/IP port serveru IPAS.
<code>void Start()</code>		Inicializuje připojení k serveru IPAS. Autentifikaci uživatele je třeba provést pomocí příkazu <code>Login()</code> .
<code>void Stop()</code>		Ukončí připojení k serveru IPAS.
<code>void Execute(Command cmd)</code>		Spustí libovolný příkaz serveru IPAS reprezentovaný instancí třídy příkazu. V případě, že volání proběhlo úspěšně (nedošlo k vyvolání výjimky), je objekt příkazu vyplněn požadovanými informacemi.

Příklady

Následující část uvádí několik jednoduchých příkladů, jak použít knihovnu Ignum Kernel pro komunikaci se serverem IPAS2.

Přihlášení k serveru IPAS

```
using Ignum.Kernel;

class IgnumKernelDemo
{
    [STAThread]
    static void Main(string[] args)
    {
        DirectSession session = new DirectSession("offline-debug.bind.ignum.cz", 5155);
        LoginUser login = new LoginUser("slon", "aaaa");
        session.Start(); // připojení k serveru
        session.Execute(login); // provedení příkazu login
        ...
        session.Stop();
    }
}
```

Volání příkazu IPAS `GetDomainInfo`

```
using System;
using Ignum.Kernel;
```

```
using Ignum.Kernel.Commands;
using Ignum.Kernel.Commands.WhoIs;

class IgnumKernelDemo
{
    [STAThread]
    static void Main(string[] args)
    {
        DirectSession session = new DirectSession("offline.core.ignum.cz", 5155);
        session.Start(); // připojení k serveru

        GetDomainInfo domainInfo = new GetDomainInfo("domena.cz");
        session.Execute(domainInfo);
        DomainInfo info = domainInfo.DomainInfo;
        Console.WriteLine(info.Name);
        Console.WriteLine(info.Description);
        Console.WriteLine(info.CreatedBy);
        Console.WriteLine(info.AdminRRID);

        session.Stop();
    }
}
```

Volání příkazu IPAS Query

```
using System;
using Ignum.Kernel;
using Ignum.Kernel.Commands;
using Ignum.Kernel.Commands.WhoIs;

class IgnumKernelDemo
{
    [STAThread]
    static void Main(string[] args)
    {
        DirectSession session = new DirectSession("offline-debug.bind.ignum.cz", 5155);
        LoginUser login = new LoginUser("slon", "aaaa");
        session.Start(); // připojení k serveru
        session.Execute(login); // provedení příkazu login
        // 10 nejdříve expirujících domén pro account 44
        Query query = new Query("350", new QueryParam[] { new QueryParam("AccID", 44) });
        session.Execute(query);

        RowList rows = query.RowList;

        for (int i = 0; i < rows.Count; i++)
        {
            Row r = rows[i];

            Console.WriteLine(r["name"] + " " + r["valid_to"]);
        }

        session.Stop();
    }
}
```

Přehled změn v knihovně

Tabulka uvádí přehled starších úprav knihoven. Aktuální informace o změnách jsou dostupné v souboru Changelog.txt distribuovaném s knihovnou.

<i>Verze</i>	<i>Změna</i>	<i>Detail</i>
2.0.2487.25365		Úpravy a implementace requestů pro komunikaci se serverem IPAS2. Některé požadavky byly rozšířeny o funkcionality umožňující jejich provádění i pro domény jiných TLD než .CZ.

Obecný popis struktury XML příkazů

Hlavním cílem této části dokumentu je popis struktury konkrétních XML příkazů pro komunikaci se serverem IPAS. Pokud používáte některou z námi dodávaných knihoven, určitě velice snadno najdete souvislost mezi popisem příkazů ve formátu XML a voláními příslušné knihovny. Pokud v jednotlivých ukázkách XML není uvedeno jinak, jsou popisované položky považovány za povinné.

Příkazy pro zjišťování informací

Příkazy uváděné v této části slouží ke zjišťování různých informací týkajících se jak uživatele, tak domén či kontaktů.

Kontrola dostupnosti domény

Ke kontrole dostupnosti domény slouží příkaz CheckDomain. Tento příkaz je možné použít i bez přihlášení uživatele k serveru IPAS. Chování příkazu je zcela odlišné od příkazu pro zjištění detailních informací o doméně. Pro ověření dostupnosti domény je tedy možné použít pouze tento příkaz:

```
<Commands>
  <Command>
    <CheckDomain>
      <!-- nazev domeny - povinne alespon jedna polozka -->
      <Name>test.com</Name>
      <!-- dalsi testovana domena -->
      <Name>test.net</Name>
    </CheckDomain>
  </Command>
</Commands>
```

Server IPAS odpoví pomocí XML s touto strukturou, odpověď na dostupnost domény je v atributu Available:

```
<Results Count="1" ExecutionTime="516">
  <Result ID="" ExecutionTime="493">
    <Status>
      <!-- nutne kontrolovat - udava, zda prikaz probehl -->
      <!-- uspesne, 0 znaci uspech -->
      <Code>0</Code>
      <Description>Command completed successfully.</Description>
    </Status>
    <!-- odpoved na prikaz -->
    <CheckDomain>
      <!-- popis jednotlivych testovanych domenovych jmen -->
      <!-- pokud atribut False, domena neni dostupna, pokud -->
      <!-- True, domena je dostupna -->
      <Name Available="False" TldID="2">test.com</Name>
      <Name Available="False" TldID="3">test.net</Name>
    </CheckDomain>
  </Result>
</Results>
```

Informace o doméně

Ke zjištění detailních informací o již registrované doméně slouží příkaz GetDomainInfo (případně GetDomainInfoByID). Tento příkaz lze obvykle použít pouze na domény, které byly registrovány přes Ignium. Z tohoto důvodu není možné příkaz použít pro testování dostupnosti domén. k tomuto účelu je nezbytné použít příkaz CheckDomain.

Rozdíl mezi příkazem GetDomainInfo a GetDomainInfoByID je v tom, že příkaz GetDomainInfoByID provede v případě potřeby import domény do systému IPAS a přiřadí ji mezi domény, které vidí uživatel ve svém seznamu. Struktura příkazu je následující:

```
<Commands>
  <Command>
```

```
<GetDomainInfo>
  <!-- nazev domeny -->
  <Domain>voloda.eu</Domain>
</GetDomainInfo>
</Command>
</Commands>
```

nebo

```
<Commands>
  <Command>
    <GetDomainInfoByID>
      <!-- nazev domeny -->
      <DomainID>123</DomainID>
    </GetDomainInfo>
  </Command>
</Commands>
```

Server IPAS vrací XML s následující strukturou. Pro jednotlivá TLD se může tato struktura mírně lišit v závislosti na dostupnosti jednotlivých informací:

```
<Results Count="1" ExecutionTime="76">
  <Result ID="" ExecutionTime="71">
    <Status>
      <!-- prikaz probehl uspesne -->
      <Code>0</Code>
      <Description>Command completed successfully.</Description>
    </Status>
    <DomainInfo>
      <!-- nazev dotazovane domeny -->
      <Domain>voloda.eu</Domain>
      <!-- systemove ID pro TLD, do ktereho domena patri -->
      <TldID>118</TldID>
      <!-- typ TLD -->
      <TldType>Synchronized</TldType>
      <!-- ID uctu IPAS, ktere je platcem domeny -->
      <PayerID>43334</PayerID>
      <!-- RRID kontaktu vlastnika domeny -->
      <OwnerRRID>c1516306</OwnerRRID>
      <!-- RRID kontaktu registratora (vzdy kontakt IGNU) -->
      <RegRRID>c38046</RegRRID>
      <!-- priznak True/False, zda je domena registrovana pres IGNU -->
      <OwnedByIgunm>True</OwnedByIgunm>
      <!-- datum registrace domeny -->
      <Created>15.06.2006 11:08:49</Created>
      <!-- RRID kontaktu registratora, ktery domenu zaregistroval -->
      <CreatedBy>t000415</CreatedBy>
      <!-- datum zmeny domeny -->
      <Updated>05.12.2006 11:31:09</Updated>
      <!-- RRID kontaktu registratora, ktery domenu zaregistroval -->
      <UpdatedBy>t000415</UpdatedBy>
      <!-- datum expirace domeny, zahrnuje pripadne virtualni prodlouzeni domeny -->
      <ValidTo>15.06.2010 00:00:00</ValidTo>
      <!-- datum realne expirace domeny v CR -->
      <RealValidTo>15.06.2007 11:08:49</RealValidTo>
      <!-- RRID administrativniho kontaktu -->
      <AdminRRID>P-123</AdminRRID>
      <!-- RRID technickeho kontaktu -->
      <TechRRID>P-123</TechRRID>
      <!-- nameservery evidovane u domeny -->
      <AttachedNServer>
        <!-- nazev nameserveru -->
        <Name>ns1.ignum.com</Name>
```

```
<!-- typ nameserveru P - primarni, s - sekundarni -->
<Type>P</Type>
</AttachedNServer>
<AttachedNServer>
  <Name>ns2.ignum.cz</Name>
  <Type>P</Type>
</AttachedNServer>
<!-- v pripade EU domeny seznam kontaktu a jejich roli -->
<AttachedContact>
  <RRID>c38046</RRID>
  <Type>billing</Type>
</AttachedContact>
<AttachedContact>
  <RRID>c1541291</RRID>
  <Type>onsite</Type>
</AttachedContact>
<AttachedContact>
  <RRID>c35855</RRID>
  <Type>tech</Type>
</AttachedContact>
<!-- status domeny, muze se lisit dle jednotlivych TLD -->
<Status>ok</Status>
</DomainInfo>
</Result>
</Results>
```

Informace o kontaktu

Ke zjištění informací o kontaktu slouží příkaz `GetContactInfo` (případně `GetContactInfoByID`). Příkaz zjišťuje informace o kontaktu přímo v centrálním registru.

Obdobně jako při zjišťování informací o doméně je rozdíl mezi příkazy `GetContactInfo` a `GetContactInfoByID` v tom, že příkaz `GetContactInfoByID` v případě potřeby naimportuje kontakt do systému IPAS a přidá ho do seznamu kontaktů přihlášeného uživatele. Struktura příkazu je následující:

```
<Commands>
  <Command>
    <GetContactInfo>
      <!-- nazev domeny - povinne alespon jedna polozka -->
      <RRID>ROB</RRID>
      <TldID>1</TldID>
    </GetContactInfo>
  </Command>
</Commands>
```

Odpověď serveru IPAS má níže uvedenou strukturu, která se může mírně lišit v závislosti na TLD, pro které je kontakt použit.

```
<Results Count="1" ExecutionTime="716">
  <Result ID="" ExecutionTime="710">
    <Status>
      <!-- prikaz probehl uspesne -->
      <Code>0</Code>
      <Description>Command completed successfully.</Description>
    </Status>
    <ContactInfo>
      <!-- RRID kontaktu -->
      <RRID>ROB</RRID>
      <!-- TLD pro ktere je mozne kontakt pouziti -->
      <TldID>1</TldID>
      <!-- jmeno kontaktu -->
      <FirstName>Robert</FirstName>
      <!-- prijmeni kontaktu -->
```

```
<LastName>Prokes</LastName>
<!-- priznak True/False zda se jedna o organizaci -->
<Individual>True</Individual>
<!-- priznak True/False zda ma kontakt nastaveno heslo -->
<WithPassword>True</WithPassword>
<!-- email kontaktu -->
<EMail>robert.prokes@ignum.cz</EMail>
<!-- telefonni kontakt -->
<Phone>+420.296332211</Phone>
<!-- fax -->
<Fax>+420.296332222</Fax>
<NotifyType>notify</NotifyType>
<Notify>robert.prokes@ignum.cz</Notify>
<Address>
  <Street>Thamova 18</Street>
  <City>Praha 8</City>
  <ZIP>11668</ZIP>
  <Country>CZ</Country>
  <CountryID>203</CountryID>
</Address>
<Created>02.10.2003 09:35:00</Created>
<CreatedBy>REG-CZNIC</CreatedBy>
<RegRRID>REG-IGNUM</RegRRID>
<Updated>01.06.2004 10:15:00</Updated>
<Status>linked</Status>
</ContactInfo>
</Result>
</Results>
```

Informace o virtuálním kontaktu

Příkaz `GetVirtualContactInfo` vrací detailní informace o virtuálním kontaktu, pokud je to možné. Informace zde obsažené odpovídají aktuálně evidované sadě dat virtuálního kontaktu. V některých případech mohou být tyto informace rozšířeny o doplňkové informace týkající se reálných kontaktů vytvořených na základě této sady dat. Příkaz má tuto strukturu:

```
<Commands>
  <Command>
    <GetVirtualContactInfo>
      <!-- ID virtualního kontaktu -->
      <VirtualContactID>10</VirtualContactID>
      <!-- volitelný priznak True/False, zda zobrazovat pouze platná mapování -->
      <!-- na RRID, defaultně True -->
      <OnlyValidMappings>True</OnlyValidMappings>
      <!-- volitelné heslo umožňující získat detailní informace -->
      <!-- i o kontaktu, který má nastaveno FullInfo = False -->
      <Password>123</Password>
    </GetVirtualContactInfo>
  </Command>
</Commands>
```

Odpověď serveru má následující strukturu (ve které může dojít k drobným odchylkám, pokud např. údaje nejsou k dispozici apod.):

```
<Results Count="1" ExecutionTime="13450">
  <Result ID="" ExecutionTime="13424">
    <Status>
      <!-- prikaz probehlo uspesne - kod 0 -->
      <Code>0</Code>
      <Description>Command completed successfully.</Description>
    </Status>
    <VirtualContactInfo>
      <!-- ID kontaktu v DB -->
      <ID>40</ID>
```

```
<!-- UID kontaktu (obdoba RRID) -->
<UID>IGNUM-V409ZE</UID>
<!-- jmeno -->
<FirstName>Jan</FirstName>
<!-- prijmeni-->
<LastName>Novak</LastName>
<!-- email -->
<EMail>jan@novak.cz</EMail>
<!-- telefon -->
<Phone>+420.602222921</Phone>
<!-- priznak True/False, zda ma kontakt heslo -->
<PasswordValid>False</PasswordValid>
<!-- priznak True/False, zda se jedna o organizaci -->
<Individual>False</Individual>
<!-- priznak, zda je povoleno zobrazovani uplnych informaci -->
<FullInfo>True</FullInfo>
<!-- nazev organizace, pokud se jedna o firmu -->
<Organization>Pears Health Cyber, s.r.o.</Organization>
<!-- adresa kontaktu -->
<Address>
  <!-- ID adresy v databazi -->
  <AddressID>55176</AddressID>
  <!-- ulice -->
  <Street>Soběslavská 23</Street>
  <!-- mesto -->
  <City>Praha 3</City>
  <!-- PSC -->
  <ZIP>13000</ZIP>
  <!-- zkratka země -->
  <Country>CZ</Country>
  <!-- ID zeme -->
  <CountryID>203</CountryID>
  <!-- typ adresy -->
  <Type>place</Type>
  <!-- defaultni adresa -->
  <Default>True</Default>
</Address>
<!-- uzivatelske ucty systemu IPAS, které mají -->
<!-- povolenu pracikontaktem -->
<AccountID>39962</AccountID>
<!-- prehled vytvorenych vazeb na skutecne kontakty -->
<RegistryMappings>
  <!-- popis jednotlivych mapovani -->
  <VirtualContactMapping>
    <!-- ID mapovani v tabulce -->
    <VirtualContactMappingID>20</VirtualContactMappingID>
    <!-- RRID kontaktu v CR -->
    <RRID>P-VAF36</RRID>
    <!-- Stav kontaktu (smazan, synchronizovan, je treba -->
    <!-- synchronizovat, probiha registrace) -->
    <State>Synchronized</State>
    <!-- TLD, pro ktere je registrace provedena -->
    <TldID>111</TldID>
    <!-- Typ kontaktu (lisi se dle TLD) -->
    <Type>0</Type>
    <!-- prehled vseh TLD, pro která je mapovani pouzitelne -->
    <ValidForTLD>
      <!-- jednotlivá TLD, pro která je mapovani platne -->
      <TldID>10</TldID>
      <TldID>14</TldID>
    </ValidForTLD>
  </VirtualContactMapping>
</RegistryMappings>
```

```
</RegistryMappings>  
</VirtualContactInfo>  
</Result>  
</Results>
```

Načtení dat z databáze pomocí předdefinovaného dotazu

Příkaz Query slouží ke spuštění předdefinovaného parametrizovaného dotazu přímo v databázi systému IPAS. Ke spuštění je třeba znát kód příkazu (základní dotazy jsou uvedeny v přehledové tabulce, v případě potřeby je možné přidat další kódy). Pokud dotaz vyžaduje další parametry, je třeba je uvést v rámci tagu Params. V případě potřeby příkaz umožňuje také stránkování (je třeba zadat parametry PageItemCount a Page). Příkaz má tuto strukturu:

```
<Commands>  
  <Command ID="0">  
    <Query>  
      <!-- ID dotazu -->  
      <ID function="1">QueryCodeToID("LIST_ACCOUNT_ORDERS")</ID>  
      <!-- volitelná položka udávající počet radku na stránku -->  
      <PageItemCount>10</PageItemCount>  
      <!-- volitelná položka udávající stránku, kterou chce uživatel zobrazit -->  
      <Page>4</Page>  
      <!-- jednotlivé parametry příkazu -->  
      <Params>  
        <!-- příkaz vyžaduje parametr ORDER a jeho hodnota bude date -->  
        <ORDER>date</ORDER>  
      </Params>  
    </Query>  
  </Command>  
</Commands>
```

Odpověď serveru IPAS má následující strukturu:

```
<Results Count="1" ExecutionTime="3210">  
  <Result ID="0" ExecutionTime="3182">  
    <!-- nacteny vety vctne informace o jejich poctu -->  
    <Rows Count="81">  
      <!-- hlavicka - popisuje sloupce ve vystupu -->  
      <Header>  
        <!-- popis jednotlivych sloupcu (datovy typ a nizev) -->  
        <Field Type="Int">ID</Field>  
        <Field Type="Int">ACCOUNT_ID</Field>  
        <Field Type="Int">SERVICE_ID</Field>  
        <Field Type="Int">TYPE</Field>  
        <Field Type="Int">SUBORDER_ID</Field>  
      </Header>  
      <!-- jednotlivé nactené vety (včetně pořadí) -->  
      <Row Pos="0">  
        <!-- hodnoty jednotlivých sloupců -->  
        <ID>333108</ID>  
        <ACCOUNT_ID>39962</ACCOUNT_ID>  
        <SERVICE_ID>78</SERVICE_ID>  
        <TYPE>0</TYPE>  
        <SUBORDER_ID />  
      </Row>  
      <Row Pos="1">  
        <ID>333109</ID>  
        <ACCOUNT_ID>39962</ACCOUNT_ID>  
        <SERVICE_ID>78</SERVICE_ID>  
        <TYPE>0</TYPE>  
        <SUBORDER_ID />  
      </Row>  
    </Rows>  
  </Status>
```

```
<!-- prikaz probehlo uspesne, kod 0 -->
<Code>0</Code>
<Description>Command completed successfully.</Description>
</Status>
</Result>
</Results>
```

Ostatní domény

Mezi ostatní domény se řadí všechna TLD neuvedená jinde. Jejich registrace probíhá s využitím virtuálních kontaktů jakožto vlastníka domény. Tyto domény poskytují pouze omezené množství příkazů, nepodporují aktualizaci údajů vlastníka domény atp. Konkrétně mezi tato TLD patří např. domény SK.

Registrace virtuálního kontaktu

Registrace virtuálního kontaktu je bezplatná objednávková služba s kódem Contact.Virtual.Create. Virtuální kontakty slouží v tuto chvíli zejména jako sada dat používaná pro identifikaci kontaktů v roli vlastníka domény, nebo v některých případech v roli administrativního kontaktu. Příkaz má tuto strukturu:

```
<Commands>
<Command>
<OrderService>
  <!-- kod objednavkove sluzby - registrace virt. Kontaktu -->
  <ServiceCode>Contact.Virtual.Create</ServiceCode>
  <!-- volitelne ID kontaktu (RRID), pokud neuvedeno -->
  <!-- je vygenerovano serverem IPAS -->
  <UID>IGNUM-V47925J</UID>
  <!-- volitelny priznak True/False, zda se jedna o firmu -->
  <!-- defaultne False -->
  <Individual>True</Individual>
  <!-- jmeno kontaktni osoby -->
  <FirstName>Jan</FirstName>
  <!-- prijmeni kontaktni osoby -->
  <LastName>Novak</LastName>
  <!-- priznak True/False, zda server IPAS vraci uplne informace -->
  <!-- o virtualnim kontaktu bez znalosti hesla -->
  <FullInfo>True</FullInfo>
  <!-- telefonicky kontakt v naznacenenem tvaru -->
  <Phone>+420.123123123</Phone>
  <!-- volitelny Fax v naznacenenem tvaru -->
  <Fax>+420.123123123</Fax>
  <!-- mailovy kontakt -->
  <EMail>jan@novak.cz</EMail>
  <!-- volitelne heslo urcene ke sdileni kontaktu -->
  <!-- pripadne k zobrazeni detailnich informace -->
  <Password>Ahoj</Password>
  <!-- volitelne DIC kontaktu -->
  <DIC>CZ1234</DIC>
  <!-- volitelne IC kontaktu -->
  <IC>123</IC>
  <!-- adresa kontaktu -->
  <CreateAddress>
    <!-- ulice a CP -->
    <Street>Ulice 123</Street>
    <!-- Mesto -->
    <City>Praha</City>
    <!-- PSC -->
    <ZIP>11150</ZIP>
    <!-- volitelne ID zeme, defaultne 203 - CZ -->
    <CountryID>203</CountryID>
  </CreateAddress>
</OrderService>
```

```
</Command>  
</Commands>
```

Změna údajů virtuálního kontaktu

Změna údajů virtuálního kontaktu je bezplatnou objednávkovou službou. Vzhledem k tomu, že změnou údajů virtuálního kontaktu se ve skutečnosti modifikuje pouze sada dat sloužící k vlastní registraci domény, nedochází automaticky provedením této změny ke změně odpovídajících reálných kontaktů v jednotlivých centrálních registrech. Hlavním důvodem je to, že ne všechny centrální registry tuto funkci podporují. Příkaz má následující strukturu.

```
<Commands>  
  <Command>  
    <OrderService>  
      <!-- kod objednavkove sluzby - registrace virt. Kontaktu -->  
      <ServiceCode>Contact.Virtual.Update</ServiceCode>  
      <!-- volitelne nove jmeno kontaktni osoby (lze menit pouze pokud se jedna -->  
      <!-- o kontakt organizace -->  
      <FirstName>Jan</FirstName>  
      <!-- volitelne nove prijmeni kontaktni osoby (lze menit pouze pokud se jedna -->  
      <!-- o kontakt organizace -->  
      <LastName>Novak</LastName>  
      <!-- volitelny priznak True/False, zda server IPAS vraci uplne informace -->  
      <!-- o virtualnim kontaktu bez znalosti hesla -->  
      <FullInfo>True</FullInfo>  
      <!-- volitelny novy telefonicky kontakt v naznacenenem tvaru -->  
      <Phone>+420.123123123</Phone>  
      <!-- volitelny novy Fax v naznacenenem tvaru -->  
      <Fax>+420.123123123</Fax>  
      <!-- volitelny novy mailovy kontakt -->  
      <EMail>jan@novak.cz</EMail>  
      <!-- volitelne nove heslo urcene ke sdileni kontaktu -->  
      <!-- pripadne k zobrazeni detailnich informace -->  
      <Password>Ahoj</Password>  
      <!-- volitelne nove DIC kontaktu -->  
      <DIC>CZ1234</DIC>  
      <!-- volitelne nove IC kontaktu -->  
      <IC>123</IC>  
      <!-- volitelna zmena adresy kontaktu -->  
      <UpdateAddress>  
        <!-- nova ulice a CP -->  
        <Street>Ulice 123</Street>  
        <!-- nove Mesto -->  
        <City>Praha</City>  
        <!-- nove PSC -->  
        <ZIP>11150</ZIP>  
        <!-- nove volitelne ID zeme, defaultne 203 - CZ -->  
        <CountryID>203</CountryID>  
      </UpdateAddress>  
    </OrderService>  
  </Command>  
</Commands>
```

Registrace ostatních národních domén

Registrace je realizována jako placená objednávková služba s kódem Domain.XXX.Create.Y, kde XXX je nahrazeno příslušným názvem TLD (v případě, že TLD je ve tvaru ORG.UK, název TLD se transformuje do tvaru ORG_UK) a Y udává délku registrace v letech (např. Domain.SK.Create.1 pro registraci SK domény na 1 rok). Registrace jinde neuvedených ostatních národních domén probíhá s využitím virtuálního kontaktu v roli vlastníka. Zpracování objednávek na tyto domény neprobíhá většinou v reálném čase a pro některá TLD je možné, že naší technickou podporou budou vyžádány další údaje potřebné pro registraci. Příkaz má tuto strukturu:

```
<Commands>
<Command>
  <OrderService>
    <!-- Kod objednavkove sluzby - registrace SK domeny na 1 rok -->
    <ServiceCode>Domain.SK.Create.1</ServiceCode>
    <Name>test.sk</Name>
    <!-- ID virtualniho kontaktu vlastnika - mozne pouzit funkci -->
    <!-- <OwnerID>10</OwnerID> -->
    <OwnerID function="1">
    <!-- pripojeny nameserver - povinne jeden primarni a alespon 2 sekundarni -->
    <AttachNServer>
      <!-- ID DNS serveru, je mozne zjistit pomoci query -->
      <ID>1</ID>
      <!-- Typ DNS - P - primarni, S - sekundarni -->
      <Type>P</Type>
    </AttachNServer>
    <AttachNServer>
      <ID>2</ID>
      <Type>S</Type>
    </AttachNServer>
    <!-- volitelny priznak, zda se ma generovat vyzva k platbe pri expiraci -->
    <!-- domeny, defaultne True -->
    <AutoRenew>True</AutoRenew>
  </OrderService>
</Command>
</Commands>
```

Prodloužení ostatních národních domén

Prodloužení ostatních národních domén je placená objednávková služba s kódem ve tvaru Domain.XXX.Renew.Y, kde XXX je zkratka TLD (v případě, že TLD je ve tvaru ORG.UK, název TLD se transformuje do tvaru ORG_UK) a Y udává délku prodloužení v letech (např. Domain.ORG_UK.Renew.2 pro prodloužení domény s koncovkou ORG.UK o 2 roky). Obvykle je možné prodloužení o 1 rok (vyjimečně na 2 roky). Příkaz má tuto strukturu:

```
<Commands>
<Command>
  <OrderService>
    <!-- kod objednavkove sluzby - prodlouzeni COM domeny o 2 roky -->
    <ServiceCode>Domain.SK.Renew.1</ServiceCode>
    <!-- nazev domeny pokud je nezname její ID, jinak ID -->
    <DomainID function="1">DomainNameToID("test.sk")</DomainID>
    <!-- <DomainID>123</DomainID> -->
  </OrderService>
</Command>
</Commands>
```

Popis odpovědí od serveru IPAS

Server IPAS na každý zasláný příkaz zasílá odpověď ve formátu XML s níže uvedenou strukturou. V případě, že je v jednom požadavku odesláno více XML příkazů, jsou tyto příkazy prováděny za sebou v pořadí uvedeném v požadavku.

```
<Results Count="1" ExecutionTime="629">
  <!-- Odpoved na prikaz, ID odpovida ID v prikazu -->
  <Result ID="" ExecutionTime="624">
    <!-- popis navratoveho kodu pozadavku -->
    <Status>
      <!-- ciselny kod, 0 udava uspesne dokonceni, jinak -->
      <!-- doslo k chybe -->
      <Code>0</Code>
      <!-- textovy popis chyby -->
      <Description>Command completed successfully.</Description>
      <!-- nepovinne ID posledniho zaznamu v DB. V pripade objednavkove -->
      <!-- sluzby udava ID objednavky apod. -->
```

```
<LastID>390515</LastID>
</Status>
<!-- nepovinne udaje o dalsi vazbe -->
<Link>
  <!-- typ vazby -->
  <Type>Domain</Type>
  <!-- ID typu vazby -->
  <TypeID>103</TypeID>
  <!-- ID provazane vety v databazi - napr. ID -->
  <!-- kontaktu, ID virtualniho kontaktu, ID domeny apod. -->
  <ID>90681</ID>
</Link>
</Result>
</Results>
```

Doména CZ a ENUM

Tato část popisuje registraci, správu a transfery kontaktů, registraci, správu a transfery sad jmenných serverů (nssetů) a registraci, správu a transfery domén CZ a ENUM¹ a validace a revalidace ENUM domén.

Dokumentace se týká nově zaváděného registračního systému využívajícího EPP protokol², který bude nasazen sdružením CZ.NIC od 1.10.2007 pro registraci CZ domén a v současné době je využíván pro registraci domén ENUM. Oproti původnímu systému pro registraci CZ domén dochází k těmto hlavním změnám:

- Dochází ke zrušení subjektů a jejich nahrazení přímo kontakty.
- Není třeba explicitně vyjadřovat souhlas s pravidly registrace domén. Za souhlas s pravidly se považuje již např. samotná registrace domény.
- Nameservery se nepřisuzují k doméně přímo, využívají se tzv. NSSETy, které obsahují informace o všech nameserverech u domény. NSSET je samostatný objekt v CR se svým správcem (technické kontakty).
- Každý objekt v CR má svého určeného registrátora, který jediný může provádět jeho změny.

V novém registračním systému budou pro obě uvedená TLD evidovány společné kontakty a NSSETy. Odlišnost mezi TLD spočívá v podstatě pouze v tom, že u domén ENUM je třeba provádět tzv. validaci vlastníka domény.

Registrace kontaktu

K registraci nového kontaktu slouží objednávková služba s kódem Contact.CZNIC.Create. Kontakt nemá definovanou žádnou specifickou roli a může proto být následně použit v roli vlastníka domény, v roli administrativního kontaktu domény i v roli technického kontaktu v rámci NSSETu. Struktura příkazu je následující:

```
<Commands>
  <Command id="0">
    <OrderService>
      <!-- kodu sluzby - registrace CZ kontaktu -->
      <ServiceCode>Contact.CZNIC.Create</ServiceCode>
      <!-- volitelny nazev organizace -->
      <Organization>My Org, a.s.</Organization>
      <!-- jmeno -->
      <FirstName>Jan</FirstName>
      <!-- prijmeni -->
      <LastName>Novak</LastName>
      <!-- volitelne RRID (handle) kontaktu,
      pokud chybi, generuje IPAS automaticky -->
      <RRID>NOVAK</RRID>
      <!-- Email kontaktu -->
      <EMail>jan.novak@domena.cz</EMail>
```

1 Pojmem ENUM se rozumí doména 0.2.4.E164.ARPA.

2 Tzv. DSDng – DSD nové generace.

```
<!-- volitelny telefon kontaktu -->
<Phone>+420.123123123</Phone>
<!-- volitelny fax kontaktu -->
<Fax>+420.123123123</Fax>
<!-- volitelny e-mail pro notifikaci -->
<NotifyEMail>jan.novak@domena.cz</NotifyEMail>
<!-- volitelne DIC firmy -->
<DIC>CZ123123</DIC>
<!-- volitelna rozsirena identifikace kontaktu -->
<Ident>
  <!-- typ identifikace: op/passport/mpsv/ico/birthday -->
  <Type>op</Type>
  <!-- ID podle typu -->
  <ID>EKJ123</ID>
</Ident>
<!-- adresa kontaktu -->
<CreateAddress>
  <!-- ulice -->
  <Street>Válečná 1</Street>
  <!-- mesto -->
  <City>Prague</City>
  <!-- PSC -->
  <ZIP>12300</ZIP>
  <!-- zeme (203 = Czech Republic) -->
  <CountryID>203</CountryID>
</CreateAddress>
<!-- Volitelny seznam polozek, které budou verejne -->
<!-- viditelne ve WhoIs. Pokud polozka neni uvedena -->
<!-- hodnota vseh prvku je False. Pokud nektery -->
<!-- prvek chybi, je jeho hodnota take False (nezobrazuje se) -->
<Disclose>
  <EMail>True</EMail>
  <Phone>True</Phone>
  <Fax>True</Fax>
  <DIC>True</DIC>
  <Ident>True</Ident>
  <NotifyEMail>True</NotifyEMail>
</Disclose>
</OrderService>
</Command>
</Commands>
```

Změna kontaktu

Ke změně kontaktu slouží objednávková služba Contact.CZNIC.Update. Změnu kontaktu je nutné autorizovat. Pro změnu názvu organizace nebo jména a příjmení v případě, kdy se nejedná o organizaci, je nutné autorizovat změnu notářsky. Ostatní změny je možné autorizovat e-mailem nebo faxem. Bez autorizace nebude objednávka vyřízena. Struktura příkazu je následující:

```
<Commands>
<Command id="0">
  <OrderService>
    <!-- kod sluzby - zmena CZ kontaktu -->
    <ServiceCode>Contact.CZNIC.Update</ServiceCode>
    <!-- ID kontaktu, lze pouzit funkci ContactRRIDToID -->
    <ContactID>123</ContactID>
    <!-- volitelny nazev organizace -->
    <Organization>My Org, a.s.</Organization>
    <!-- volitelna zmena jmena, pokud uvedeno, musi -->
    <!-- byt uvedeno tez LastName -->
    <FirstName>Jan</FirstName>
    <!-- volitelna zmena prijmeni, pokud uvedeno, musi -->
```

```
<!-- byt uvedeno tez FirstName -->
<LastName>Novak</LastName>
<!-- volitelna zmena Emailu kontaktu -->
<EMail>jan.novak@domena.cz</EMail>
<!-- volitelny telefon kontaktu -->
<Phone>+420.123123123</Phone>
<!-- volitelny fax kontaktu -->
<Fax>+420.123123123</Fax>
<!-- volitelny e-mail pro notifikaci -->
<NotifyEMail>jan.novak@domena.cz</NotifyEMail>
<!-- volitelne DIC firmy -->
<DIC>CZ123123</DIC>
<!-- volitelna rozsirena identifikace kontaktu -->
<Ident>
  <!-- typ identifikace: op/passport/mpsv/ico/birthday -->
  <Type>op</Type>
  <!-- ID podle typu -->
  <ID>EKJ123</ID>
</Ident>
<!-- volitelna nova adresa kontaktu -->
<CreateAddress>
  <!-- ulice -->
  <Street>Válečná 1</Street>
  <!-- mesto -->
  <City>Prague</City>
  <!-- PSC -->
  <ZIP>12300</ZIP>
  <!-- zeme (203 = Czech Republic) -->
  <CountryID>203</CountryID>
</CreateAddress>
<!-- Volitelny seznam polozek, které budou verejne -->
<!-- viditelne ve WhoIs. Pokud polozka neni uvedena -->
<!-- nedochazi ke zmene. Pokud nektery prvek -->
<!-- prvek chybi, je jeho hodnota nastavena na False (nezobrazuje se) -->
<Disclose>
  <EMail>True</EMail>
  <Phone>True</Phone>
  <Fax>True</Fax>
  <DIC>True</DIC>
  <Ident>True</Ident>
  <NotifyEMail>True</NotifyEMail>
</Disclose>
<!-- autorizace zmeny -->
<Authenticate>
  <!-- zpusob autorizace zmeny - Notarial/Fax/EMail -->
  <Type>EMail</Type>
  <!-- ID kontaktu - stejna hodnota jako ContactID pro -->
  <!-- ktere se provadi zmena -->
  <ContactID>123</ContactID>
  <!-- V pripade autorizace Faxem nebo notarsky adresa, -->
  <!-- na kterou bude zaslan predvyplneny autorizacni
  <!-- formular -->
  <EMail>jan.novak@domena.cz</EMail>
</Authenticate>
</OrderService>
</Command>
</Commands>
```

Smazání kontaktu

Ke zrušení kontaktu v registru slouží objednávková služba Contact.CZNIC.Delete. Smazání je nutné autorizovat buď faxem nebo e-mailem. Bez autorizace nebude objednávka vyřízena. Struktura příkazu je následující:

```
<Commands>
  <Command>
    <OrderService>
      <!-- kod objednavkove sluzby - smazani kontaktu -->
      <ServiceCode>Contact.CZNIC.Delete</ServiceCode>
      <!-- ID kontaktu v DB, lze pouzít funkci ContactRRIDtoID -->
      <ContactID>1</ContactID>
      <Authenticate>
        <!-- stejná hodnota jako ID kontaktu, který se rusí -->
        <ContactID>1</ContactID>
        <!-- typ autorizace - Fax, EMail -->
        <Type>Fax</Type>
        <!-- volitelná e-mailová adresa, na kterou bude zaslán předvyplněný -->
        <!-- notarský nebo faxový formulář -->
        <EMail>novak@domena.cz</EMail>
      </Authenticate>
    </OrderService>
  </Command>
</Commands>
```

Registrace NSSETu

NSSET (sada jmených serverů) je v CR samostatným objektem. Má své správce (technické kontakty), kteří ho mají právo editovat, případně ho mohou vyměnit za jiný NSSET u domén, kde je použit. Tento objekt zcela nahrazuje jednotlivé nameservery u domén. K vytvoření nového NSSETu slouží objednávková služba NSSet.CZNIC.Create. Struktura příkazu je následující:

```
<Commands>
  <Command>
    <OrderService>
      <!-- kod objednavkove sluzby - vytvoreni NSSETu -->
      <ServiceCode>NSSet.CZNIC.Create</ServiceCode>
      <!-- volitelne RRID (handle) NSSETu v CR, pokud chybi -->
      <!-- je vygenerovano automaticky IPASem -->
      <RRID>NSSID:123</RRID>
      <!-- 1 - 10 kontaktu, které budou nastaveny jako -->
      <!-- technicky spravce NSSETu -->
      <TechID>12</TechID>
      <TechID function="1">ContactRRIDtoID("CID:123", 1)</TechID>
      <!-- 2 - 9 nameserveru, které budou prirazeny k NSSETu -->
      <AttachNServer>
        <!-- ID DNS serveru, je mozne zjistit pomoci query -->
        <ID>1</ID>
        <!-- Typ DNS - P - primarni, S - sekundarni -->
        <Type>P</Type>
      </AttachNServer>
      <AttachNServer>
        <ID>2</ID>
        <Type>S</Type>
      </AttachNServer>
    </OrderService>
  </Command>
</Commands>
```

Změna NSSETu

Ke změně NSSETu slouží objednávková služba NSSet.CZNIC.Update. Každá změna musí být autorizována jedním z technických kontaktů, které jsou u NSSETu evidovány. Možné způsoby validace jsou e-mailem nebo faxem. Struktura příkazu je následující:

```
<Commands>
  <Command>
```

```
<OrderService>
  <!-- kod objednavkove sluzby - zmena NSSETu -->
  <ServiceCode>NSSet.CZNIC.Update</ServiceCode>
  <NSSetID function="1">NSSetRRIDToID("NSSID:IGNUM", 1)</NSSetID>
  <!-- 0 - 10 kontaktu, které budou nastaveny jako -->
  <!-- technicky spravce NSSETu. Pokud uveden alespon jeden, -->
  <!-- je provedena kompletni nahrada za kontakty uvedene v requestu -->
  <TechID>12</TechID>
  <!-- 0 - 9 nameserveru, které budou prirazeny k NSSETu, -->
  <!-- pokud neuveden zadny, nedojde v seznamu k zadne zmene, -->
  <!-- pokud uveden alespon jeden, dojde k nahrade vsech -->
  <AttachNServer>
    <!-- ID DNS serveru, je mozne zjistit pomoci query -->
    <ID>1</ID>
    <!-- Typ DNS - P - primarni, S - sekundarni -->
    <Type>P</Type>
  </AttachNServer>
  <AttachNServer>
    <ID>2</ID>
    <Type>S</Type>
  </AttachNServer>
  <!-- autorizace zmeny -->
  <Authenticate>
    <!-- zpusob autorizace zmeny - Fax/EMail -->
    <Type>Fax</Type>
    <!-- ID tech. kontaktu, který autorizuje zmenu -->
    <ContactID>123</ContactID>
    <!-- V pripade autorizace Faxem, na kterou bude -->
    <!-- zaslan predvyplneny autorizacni formular -->
    <EMail>jan.novak@domena.cz</EMail>
  </Authenticate>
</OrderService>
</Command>
</Commands>
```

Smazání NSSETu

Ke zrušení NSSETu v CR slouží objednávková služba NSSet.CZNIC.Delete. Zrušení NSSETu je možné autorizovat buď e-mailem nebo faxem. Struktura příkazu je následující:

```
<Commands>
  <Command>
    <OrderService>
      <!-- kod objednavkove sluzby - zruseni NSSETu -->
      <ServiceCode>NSSet.CZNIC.Delete</ServiceCode>
      <!-- ID domeny v DB, lze pouzit funkci NSSetRRIDToID -->
      <NSSetID>1</NSSetID>
      <Authenticate>
        <!-- ID tech. kontaktu pro autorizaci, lze pouzit -->
        <!-- funkci ContactRRIDToID -->
        <ContactID>234</ContactID>
        <!-- typ autorizace - Fax, EMail -->
        <Type>Fax</Type>
        <!-- volitelna e-mailova adresa, na kterou bude zaslan predvyplneny -->
        <!-- faxovy formular -->
        <EMail>novak@domena.cz</EMail>
      </Authenticate>
    </OrderService>
  </Command>
</Commands>
```

Registrace domény CZ

K registraci nové CZ domény slouží objednávková služba Domain.CZ.Create.X, kde X určuje délku registrace v letech (tj. kód Domain.CZ.Create.1 registruje doménu na 1 rok). K registraci je třeba mít vytvořen NSSET a kontakt vlastníka domény. Struktura příkazu je následující:

```
<Commands>
  <Command>
    <OrderService>
      <!-- kod objednavkove sluzby - registrace CZ domeny na 1 rok -->
      <ServiceCode>Domain.CZ.Create.1</ServiceCode>
      <!-- nazev domeny -->
      <Name>test.cz</Name>
      <!-- volitelny priznak True/False, zda domenu automaticky -->
      <!-- prodluzovat, pokud chybi, hodnota True -->
      <AutoRenew>True</AutoRenew>
      <!-- NSSET pro pouziti u domeny -->
      <NSSetID function="1">NSSetRRIDToID("NSSID:IGN-N4-2P4", 1)</NSSetID>
      <!-- ID kontaktu vlastnika domeny -->
      <RegistrantID>85053</RegistrantID>
      <!-- 0 - 10 administrativnich kontaktu, lze pouzit fci -->
      <!-- ContactRRIDToID -->
      <AdminID>85053</AdminID>
      <AdminID function="1">ContactRRIDToID("CID:123", 1)</AdminID>
    </OrderService>
  </Command>
</Commands>
```

Speciální registrace domény CZ

Speciální registrace domény CZ umožňuje vytvořit objednávku na CZ doménu, která je v expiraci a nebyla dosud prodloužena. Služba funguje tak, že se server IPAS pokusí zaregistrovat doménu v okamžiku, kdy je zrušena v CR (45 dnů po expiraci). Služba nezaručuje, že zákazník tuto doménu získá. Pokud se registrace nepodaří, jsou uhrazené prostředky za objednávku navráceny na kredit zákazníka. Kód služby je DomainSpec.CZ.Create.X, kde X udává délku registrace v letech a její parametry jsou shodné jako pro službu Domain.CZ.Create.X.

Prodloužení domény CZ a ENUM

K prodloužení registrace domény CZ (ENUM) slouží objednávková služba Domain.CZ.Renew.X (Domain.ENUM.Renew.X), kde X je doba prodloužení v letech (služba Domain.CZ.Renew.5 tedy prodlouží doménu CZ o 5 let). Maximální celková doba prodloužení domény je 10 let. Struktura příkazu je následující:

```
<Commands>
  <Command>
    <OrderService>
      <!-- kod objednavkove sluzby - prodlouzeni CZ domeny o 5 let -->
      <ServiceCode>Domain.CZ.Renew.5</ServiceCode>
      <!-- ID prodluzovane domeny -->
      <DomainID>3447</DomainID>
    </OrderService>
  </Command>
</Commands>
```

Změna domény CZ a ENUM

Ke změně údajů evidovaných u domény CZ (ENUM) slouží objednávková služba Domain.CZ.Update (Domain.ENUM.Update). Tato služba neumožňuje změnit vlastníka domény. Veškeré prováděné změny je třeba autorizovat. Změnu NSSETu smí potvrdit také technický správce NSSETu, ostatní změny smí provádět všechny kontakty evidované u domény. Bez autorizace nebude objednávka vyřízena. Struktura příkazu je následující:

```
<Commands>
```

```
<Command>
  <OrderService>
    <!-- kod objednavkove sluzby - zmena domeny -->
    <ServiceCode>Domain.CZ.Update</ServiceCode>
    <!-- ID domeny -->
    <DomainID>123</DomainID>
    <!-- volitelne ID noveho NSSETu, lze pouzit -->
    <!-- funkci NSSetRRIDToID -->
    <NSSetID>134</NSSetID>
    <!-- volitelna zmena administrativnich kontaktu -->
    <!-- pokud neni uveden zadny, ke zmene nedochazi -->
    <!-- jinak jsou nahrazeny vsechny kontakty uvedenemi -->
    <AdminID>456</AdminID>
    <AdminID>12454</AdminID>
    <!-- volitelna polozka se seznamem temporary kontaktu, -->
    <!-- ktere maji byt odstraneny -->
    <DeleteContact>
      <!-- ID temporary kontaktu, ktere ma byt odstraneno -->
      <TempID>244</TempID>
      <TempID>453</TempID>
    </DeleteContact>
    <!-- autorizace zmeny -->
    <Authenticate>
      <!-- zpusob autorizace - Email/Fax -->
      <Type>EMail</Type>
      <!-- kontakt, ktery autorizuje zmenu (vlastnik domeny, admin -->
      <!-- nebo v pripade zmeny NSSETu technicky kontakt stavajiciho -->
      <!-- NSSETu -->
      <ContactID>4535</ContactID>
      <!-- emailova adresa, na kterou bude zaslan -->
      <!-- predvyplneny faxovy formular -->
      <EMail>jan@novak.cz</EMail>
    </Authenticate>
  </OrderService>
</Command>
</Commands>
```

Změna vlastníka domény CZ

Ke změně vlastníka CZ domény slouží objednávková služba Domain.CZ.ChgOwner. Změnu je nutné autorizovat pomocí notářsky ověřeného formuláře. Bez této autorizace nebude objednávka vyřízena. Struktura příkazu je následující:

```
<Commands>
  <Command>
    <OrderService>
      <!-- kod objednavkove sluzby - zmena vlastnika domeny -->
      <ServiceCode>Domain.CZ.Delete</ServiceCode>
      <!-- ID domeny -->
      <DomainID function="1">DomainNameToID("ahoj.cz")</DomainID>
      <!-- ID noveho vlastnika domeny -->
      <RegistrantID>123</RegistrantID>
      <!-- autorizace zmeny -->
      <Authenticate>
        <!-- typ autorizace - pouze notarsky -->
        <Type>Notarial</Type>
        <!-- ID kontaktu aktualniho vlastnika -->
        <ContactID>234</ContactID>
        <!-- emailova adresa, na kterou bude zaslan -->
        <!-- predvyplneny notarsky formular -->
        <EMail>novak@domena.cz</EMail>
      </Authenticate>
    </OrderService>
```

```
</Command>  
</Commands>
```

Zrušení domény CZ a ENUM

Ke zrušení stávající registrace domény CZ (ENUM) slouží objednávková služba Domain.CZ.Delete (Domain.ENUM.Delete). Objednávku je nutné autorizovat pomocí notářsky ověřeného formuláře. Bez této autorizace nebude objednávka provedena. Struktura příkazu je následující:

```
<Commands>  
  <Command>  
    <OrderService>  
      <!-- kod objednavkove sluzby - smazani domeny -->  
      <ServiceCode>Domain.CZ.Delete</ServiceCode>  
      <!-- ID domeny v DB, lze pouzít funkci DomainNameToID -->  
      <DomainID>1</DomainID>  
      <Authenticate>  
        <!-- ID kontaktu vlastnika domeny, lze pouzít -->  
        <!-- funkci ContactRRIDToID -->  
        <ContactID>234</ContactID>  
        <!-- typ autorizace - pouze notarskym formularem -->  
        <Type>Notarial</Type>  
        <!-- volitelna e-mailova adresa, na kterou bude zaslan predvyplneny -->  
        <!-- notarsky formular -->  
        <EMail>novak@domena.cz</EMail>  
      </Authenticate>  
    </OrderService>  
  </Command>  
</Commands>
```

Registrace domény ENUM

K registraci nové ENUM domény slouží objednávková služba Domain.ENUM.Create.X, kde X určuje délku registrace v letech (tj. kód Domain.ENUM.Create.1 registruje doménu na 1 rok). K registraci je třeba mít vytvořen NSSET a kontakt vlastníka. Je třeba též zvolit způsob validace vlastníka domény – objednávka registrace domény automaticky vytvoří objednávku na validaci ENUM čísla. Před započítáním registrace domény je třeba nejprve provést validaci vlastníka domény. Struktura příkazu je téměř shodná se strukturou pro registraci domény CZ a je následující:

```
<Commands>  
  <Command>  
    <OrderService>  
      <!-- kod objednavkove sluzby - registrace CZ domeny na 1 rok -->  
      <ServiceCode>Domain.CZNIC.Create.1</ServiceCode>  
      <!-- nizev domeny -->  
      <Name>2.3.4.2.0.2.4.e164.arpa</Name>  
      <!-- volitelny priznak True/False, zda domenu automaticky -->  
      <!-- prodluzovat, pokud chybi, hodnota True -->  
      <AutoRenew>True</AutoRenew>  
      <!-- NSSET pro pouziti u domeny -->  
      <NSSetID function="1">NSSetRRIDToID("NSSID:IGN-N4-2P4", 1)</NSSetID>  
      <!-- ID kontaktu vlastnika domeny -->  
      <RegistrantID>85053</RegistrantID>  
      <!-- 0 - 10 administrativnich kontaktu, lze pouzít fci -->  
      <!-- ContactRRIDToID -->  
      <AdminID>85053</AdminID>  
      <AdminID function="1">ContactRRIDToID("CID:123", 1)</AdminID>  
      <!-- prvotni způsob validace - SMS/Notarial -->  
      <Validation>SMS</Validation>  
    </OrderService>  
  </Command>  
</Commands>
```

Změna vlastníka domény ENUM

Ke změně vlastníka ENUM domény slouží objednávková služba Domain.ENUM.ChgOwner. Změnu vlastníka je nutné autorizovat pomocí notářsky ověřeného formuláře. Bez této autorizace nebude objednávka vyřízena. Součástí změny vlastníka je též validace vlastníka nového (funkčnost je stejná jako při registraci nové domény). Struktura příkazu je následující:

```
<Commands>
  <Command>
    <OrderService>
      <!-- kod objednavkove sluzby - zmena vlastnika domeny -->
      <ServiceCode>Domain.CZ.Delete</ServiceCode>
      <!-- ID domeny -->
      <DomainID function="1">DomainNameToID("ahoj.cz")</DomainID>
      <!-- ID noveho vlastnika domeny -->
      <RegistrantID>123</RegistrantID>
      <!-- autorizace zmeny -->
      <Authenticate>
        <!-- typ autorizace - pouze notarsky -->
        <Type>Notarial</Type>
        <!-- ID kontaktu aktualniho vlastnika -->
        <ContactID>234</ContactID>
        <!-- emailova adresa, na kterou bude zaslan -->
        <!-- predvyplneny notarsky formular -->
        <EMail>novak@domena.cz</EMail>
      </Authenticate>
      <!-- zpusob validace domeny - SMS/Notarial -->
      <Validation>SMS</Validation>
    </OrderService>
  </Command>
</Commands>
```

Validace a revalidace vlastníka domény ENUM

Validace domény ENUM slouží k ověření skutečnosti, že vlastník domény je skutečným vlastníkem odpovídajícího telefonního čísla. Validace domény musí proběhnout vždy do 6 měsíců od provedení poslední validace. V systému IPAS jsou možné dva způsoby validace:

- pomocí SMS zprávy zaslané na tel. číslo registrované domény (pozor – v případě, že se jedná o pevnou linku je SMS zpráva doručena a přečtena hlasovým automatem).
- pomocí formuláře (tzv. notářská) validace – jako formulář může sloužit např. vyúčtování služeb, na kterém je uveden shodný název jako u vlastníka domény, potvrzení od poskytovatele tel. čísla apod.

Pro objednávku validace slouží objednávkové služby Domain.SMS.Validate a Domain.Notarial.Validate. První validace vlastníka domény je povinná a objednávka je vytvořena automaticky při registraci ENUM domény (případně při změně vlastníka).

K objednávce revalidace slouží služby Domain.SMS.Revalidate a Domain.Notarial.Revalidate. Objedávka na revalidaci je v systému vedena jako zaplacená až do doby, kdy je revalidace třeba. V tu chvíli se použije nejdříve vytvořená objednávka na revalidaci. Uživatel tedy může mít vytvořené a zaplacené objednávky na revalidaci na libovolně dlouhou dobu dopředu. V případě, že uživatel žádnou objednávku na revalidaci nemá k dispozici, je mu tato 14 dní před expirací aktuální validace automaticky vytvořena. Objedávky, které jsou zaplacené, ale ještě nejsou zpracovávány, lze kdykoliv stornovat, přičemž částka za objednávku se vrátí na kredit zákazníka. V případě transferu (změny určeného registrátora) se první revalidace vlastníka provádí až ve chvíli, kdy expiruje validace původní. Struktura příkazu je následující:

```
<Commands>
  <Command>
    <OrderService>
      <!-- kod objednavkove sluzby - validace domeny pomoci SMS -->
      <ServiceCode>Domain.SMS.Revalidate</ServiceCode>
```

```
<!-- ID domeny, pro kterou se objednávka používá -->
<DomainID>6543</DomainID>
</OrderService>
</Command>
</Commands>
```

Transfery kontaktů, NSSETů a domén CZ a ENUM

Každý objekt v registru má svého tzv. určeného registrátora. Pouze tento určený registrátor je oprávněn provádět změny v parametrech objektu. Při registraci NSSETu je možné využít kontakty s jiným určeným registrátorem. Stejně tak je možné při registraci domén využít jak NSSET tak kontakt, který má jiného určeného registrátora.

Všechny objednávkové služby pro transfer vyžadují znalost tzv. Auth-ID. Jedná se v podstatě o kód, kterým vlastník objektu potvrdí svoji identitu a autorizuje převod objektu. V případě, že uživatel tento kód nezná, je možné si vyžádat jeho zaslání na odpovídající e-mailovou adresu evidovanou u objektu v CR. Pokud tuto adresu není možné z nějakého důvodu použít, je možné tento kód zjistit v CZ.NICu.

Zaslání Auth-ID pro transfer kontaktu

Pro přeposlání autorizačního kódu k transferu kontaktu slouží objednávková služba Contact.CZNIC.TransferOut. Autorizační kód je přeposlán na e-mail evidovaný u kontaktu v CR. Struktura příkazu je následující:

```
<Commands>
<Command>
  <OrderService>
    <!-- kód objednávkové služby - transfer kontaktu -->
    <ServiceCode>Contact.CZNIC.TransferOut</ServiceCode>
    <!-- ID kontaktu v DB, lze použít funkci ContactRRIDToID -->
    <ContactID>1</ContactID>
  </OrderService>
</Command>
</Commands>
```

Zaslání Auth-ID pro transfer NSSETu

Pro přeposlání autorizačního kódu k transferu NSSETu slouží objednávková služba NSSet.CZNIC.TransferOut. Autorizační kód je přeposlán na kontaktní e-maily všech technických kontaktů NSSETu evidovaných v CR. Struktura příkazu je následující:

```
<Commands>
<Command>
  <OrderService>
    <!-- kód objednávkové služby - transfer kontaktu -->
    <ServiceCode>NSSet.CZNIC.TransferOut</ServiceCode>
    <!-- ID kontaktu v DB, lze použít funkci NSSetRRIDToID -->
    <NSSetID>1</NSSetID>
  </OrderService>
</Command>
</Commands>
```

Zaslání Auth-ID pro transfer domény CZ a ENUM

Pro přeposlání autorizačního kódu k transferu domény slouží objednávková služba Domain.CZ.TransferOut (případně Domain.ENUM.TransferOut pro domény ENUM). Autorizační kód je přeposlán na kontaktní e-mail vlastníka domény evidovaný v CR. Struktura příkazu je následující:

```
<Commands>
<Command>
  <OrderService>
    <!-- kód objednávkové služby - transfer kontaktu -->
    <ServiceCode>Domain.CZ.TransferOut</ServiceCode>
```

```
<!-- ID kontaktu v DB, lze použít funkci DomainNameToID -->
<DomainID>1</DomainID>
</OrderService>
</Command>
</Commands>
```

Objednávka transferu kontaktu pod IGNUm

K transferu kontaktu pod IGNUm slouží objednávková služba Contact.CZNIC.Transfer, k autorizaci transferu je nutné znát autorizační kód Auth-ID. Struktura příkazu je následující:

```
<Commands>
<Command>
  <OrderService>
    <!-- kod objednavkove sluzby - transfer kontaktu -->
    <ServiceCode>Contact.CZNIC.Transfer</ServiceCode>
    <!-- RRID kontaktu -->
    <RRID>IGNUM-01012-AS</RRID>
    <!-- autorizacni kod -->
    <AuthID>123asd</AuthID>
  </OrderService>
</Command>
</Commands>
```

Objednávka transferu NSSETu pod IGNUm

K transferu NSSETu pod IGNUm slouží objednávková služba NSSet.CZNIC.Transfer, k autorizaci transferu je nutné znát autorizační kód Auth-ID. Struktura příkazu je následující:

```
<Commands>
<Command>
  <OrderService>
    <!-- kod objednavkove sluzby - transfer NSSETu -->
    <ServiceCode>NSSet.CZNIC.Transfer</ServiceCode>
    <!-- RRID NSSETu -->
    <RRID>IGNUM-01012-AS</RRID>
    <!-- autorizacni kod -->
    <AuthID>123asd</AuthID>
  </OrderService>
</Command>
</Commands>
```

Objednávka transferu domény CZ nebo ENUM pod IGNUm

K transferu domény CZ (ENUM) pod IGNUm slouží objednávková služba Domain.CZ.Transfer (Domain.ENUM.Transfer). K autorizaci požadavku je nutné znát autorizační kód. Struktura příkazu je následující:

```
<Commands>
<Command>
  <OrderService>
    <!-- kod objednavkove sluzby - registrace generickeho kontaktu -->
    <ServiceCode>Domain.CZ.Transfer</ServiceCode>
    <!-- nazev domeny -->
    <Domain>test.cz</Domain>
    <!-- autorizacni kod -->
    <AuthID>123asd</AuthID>
    <!-- automaticky prodluzovat, nepovinný, defaultně True, - True/False -->
    <AutoRenew>True</AutoRenew>
  </OrderService>
</Command>
</Commands>
```

Informace o kontaktu

Pro zjištění informací o kontaktu slouží obecný příkaz `GetContactInfo`, případně `GetContactInfoByID`. Vracené informace jsou rozšířeny tak, aby respektovaly informace použité při vytváření kontaktu.

Informace o NSSETu

Pro zjištění informací o NSSETu slouží příkaz `GetNSSetInfo`, případně `GetNSSetInfoByID`. Vracené informace odpovídají údajům uváděným při registraci domény. Struktura příkazů je následující:

```
<Commands>
  <Command>
    <GetNSSetInfo>
      <!-- RRID NSSETu -->
      <RRID>NSSID:IGNUM</RRID>
    </GetNSSetInfo>
  </Command>
</Commands>
```

Případně:

```
<Commands>
  <Command>
    <GetNSSetInfoByID>
      <!-- ID NSSETu v tabulce NSERVER_SET -->
      <NSSetID>1</NSSetID>
    </GetNSSetInfoByID>
  </Command>
</Commands>
```

Informace o doméně CZ a ENUM

Pro zjištění informací o doméně CZ (ENUM) slouží standardní příkaz `GetDomainInfo`, případně `GetDomainInfoByID`. Vracené údaje odpovídají údajům použitým při registraci domény, v případě domény ENUM je k dispozici expirace validace vlastníka domény.

Přidání NSSETu do seznamu uživatele

Pro přidání NSSETu do seznamu uživatele slouží příkaz `AssignNSSet`. Struktura příkazu je následující:

```
<Commands>
  <Command>
    <AssignNSSet>
      <!-- RRID NSSETu -->
      <RRID>NSSID:IGNUM</RRID>
    </AssignNSSet>
  </Command>
</Commands>
```

Odebrání NSSETu ze seznamu uživatele

Pro odebrání NSSETu ze seznamu uživatele slouží příkaz `DetachNSSet`. Struktura příkazu je následující:

```
<Commands>
  <Command>
    <DetachNSSet>
      <!-- ID NSSETu -->
      <NSSetID>1234</NSSetID>
    </DetachNSSet>
  </Command>
</Commands>
```

Generické domény

Tato část popisuje registraci kontaktů pro generické domény, změnu údajů těchto kontaktů, registraci a prodlužování generických domén, jejich transfer pod správu Ignium, a změnu údajů těchto domén. Za generické domény jsou níže považována tato TLD: COM, NET, ORG, BIZ, NAME, INFO.

Kontakt slouží buď jako datová základna pro vlastníka domény (po registraci domény již neexistuje žádná zpětná vazba mezi kontaktem a vlastníkem domény) nebo jako kontakt v jiných rolích (pak po aktualizaci údajů tohoto kontaktu dochází k promítnutí změn u všech domén). V případě, že je třeba měnit údaje vlastníka domény, je vždy třeba provést editaci změnu této domény.

Registrace generického kontaktu

K registraci generického kontaktu slouží bezplatná objednávková služba s kódem Contact.JOKER.Create. Součástí příkazu je TLD, pro které je kontakt platný. Struktura celého příkazu je následující:

```
<Commands>
  <Command>
    <OrderService>
      <!-- kod objednavkove sluzby - registrace generickeho kontaktu -->
      <ServiceCode>Contact.JOKER.Create</ServiceCode>
      <!-- TLD, pro ktere je kontakt registrovan: 2 - COM, 3 - NET, 4 - ORG, -->
      <!-- 5 - BIZ, 6 - INFO. Lze zjistit pomoci Query -->
      <TldID>1</TldID>
      <!-- krestni jmeno kontaktu -->
      <FirstName>Jan</FirstName>
      <!-- prijmeni kontaktu -->
      <LastName>Novak</LastName>
      <!-- volitelny titul kontaktu -->
      <Title>Mrs.</Title>
      <!-- volitelny nazev organizace pokud se jedna o firmu -->
      <Organization>Moje firma, s.r.o.</Organization>
      <!-- volitelny priznak, zda se jedna o organizaci: True - fyz. Osoba, -->
      <!-- False organizace -->
      <Individual>False</Individual>
      <!-- email pro komunikaci -->
      <Email>jan.novak@mojefirma.cz</EMail>
      <!-- telefonicky kontakt v naznacenenem tvaru -->
      <Phone>+420.123123123</Phone>
      <!-- volitelna polozka Fax v naznacenenem tvaru -->
      <Fax>+420.123123123</Fax>
      <!-- adresa kontaktu -->
      <CreateAddress>
        <!-- jmeno pouzite v adrese -->
        <Name>Moje firma, s.r.o.</Name>
        <!-- ulice a CP -->
        <Street>Ulice 123</Street>
        <!-- Mesto -->
        <City>Praha</City>
        <!-- PSC -->
        <ZIP>11150</ZIP>
        <!-- volitelne ID zeme, defaultne 203 - CZ -->
        <CountryID>203</CountryID>
      </CreateAddress>
    </OrderService>
  </Command>
</Commands>
```

Změna generického kontaktu

Změna údajů generického kontaktu je bezplatná objednávková služba s kódem Contact.JOKER.Update. Pro dosažení

změny evidovaných údajů vlastníka domény, pak je třeba použít objednávkovou službu na změnu údajů domény. Struktura celého příkazu je následující:

```
<Commands>
  <Command>
    <OrderService>
      <!-- kod objednavkove sluzby - zmena udaju generickeho kontaktu -->
      <ServiceCode>Contact.JOKER.Update</ServiceCode>
      <!-- ID kontaktu pro který se provádí změna. Při použití interního ID není -->
      <!-- třeba používat funkci. TLD v funkci nabyvá hodnotu dle TLD kontaktu: -->
      <!-- 2 - COM, 3 - NET, 4 - ORG, -->
      <!-- 5 - BIZ, 6 - INFO. Lze zjistit pomocí Query -->
      <ContactID function="1">ContactRRIDToID("CCOM-123", 2)</ContactID>
      <!-- <ContactID>123</ContactID> -->
      <!-- křestní jméno kontaktu -->
      <FirstName>Jan</FirstName>
      <!-- příjmení kontaktu -->
      <LastName>Novak</LastName>
      <!-- volitelný titul kontaktu -->
      <Title>Mrs.</Title>
      <!-- volitelný název organizace -->
      <Organization>Moje firma, s.r.o.</Organization>
      <!-- volitelný prázek, zda se jedná o organizaci: True - fyz. osoba, -->
      <!-- False organizace -->
      <Individual>False</Individual>
      <!-- volitelný nový email pro komunikaci -->
      <Email>jan.novak@mojefirma.cz</Email>
      <!-- volitelný nový telefonický kontakt v naznačeném tvaru -->
      <Phone>+420.123123123</Phone>
      <!-- volitelný nový fax v naznačeném tvaru -->
      <Fax>+420.123123123</Fax>
      <!-- volitelná nová adresa kontaktu -->
      <UpdateAddress>
        <!-- volitelně nové jméno použité v adrese -->
        <Name>Moje firma, s.r.o.</Name>
        <!-- volitelná nová ulice a CP -->
        <Street>Ulice 123</Street>
        <!-- volitelně nové město -->
        <City>Praha</City>
        <!-- volitelně nové PSČ -->
        <ZIP>11150</ZIP>
        <!-- volitelně nové ID země, defaultně 203 - CZ -->
        <CountryID>203</CountryID>
      </UpdateAddress>
    </OrderService>
  </Command>
</Commands>
```

Registrace generické domény

K registraci generické domény slouží placená objednávková služba s kódem Domain.XXX.Create.Y. XXX je nahrazeno příslušným TLD a Y udává délku registrace v letech. Domény je možné registrovat minimálně na 1 rok, maximálně na 10 let (tj. např. Domain.COM.Create.1 pro registraci domény COM na 1 rok). Struktura celého příkazu je následující:

```
<Commands>
  <Command>
    <OrderService>
      <!-- kod objednavkove sluzby - transfer COM domeny s prodlouzenim o 1 rok -->
      <ServiceCode>Domain.COM.Create.1</ServiceCode>
      <!-- nazev domeny pokud je neznáme její ID, jinak ID -->
      <Name>test.com</Name>
      <!-- povinný kontakt vlastníka domény - slouží pouze pro získání údajů -->
```

```
<!-- pro registraci. Nema dalsi vazbu na domenu).  
<!-- 2 je ID TLD COM v DB IPAS, je mozne zjistit pomoci query -->  
<OwnerID function="1">ContactRRIDToID("CCOM-123", 2)</OwnerID>  
<!-- technicky kontakt domeny, volitelna polozka -->  
<TechID function="1">ContactRRIDToID("CCOM-124", 2)</TechID>  
<!-- administrativni kontakt domeny, volitelna polozka -->  
<AdminID function="1">ContactRRIDToID("CCOM-125", 2)</AdminID>  
<!-- pripojeny nameserver - povinne jeden primarni a alespon 2 sekundarni -->  
<AttachNServer>  
  <!-- ID DNS serveru, je mozne zjistit pomoci query -->  
  <ID>1</ID>  
  <!-- Typ DNS - P - primarni, S - sekundarni -->  
  <Type>P</Type>  
</AttachNServer>  
<AttachNServer>  
  <ID>2</ID>  
  <Type>S</Type>  
</AttachNServer>  
<!-- volitelny priznak, zda se ma generovat vyzva k platbe pri expiraci -->  
<!-- domeny, defaultne True -->  
<AutoRenew>True</AutoRenew>  
</OrderService>  
</Command>  
</Commands>
```

Transfer generické domény

Transferem domény se rozumí přesun domény pod správu Ignium. Transfer je možný pouze s prodloužením data expirace alespoň o 1 rok a jedná se proto o placenou objednávkovou službu. Pro transfer všech typů generických domén je třeba zadat tzv. Auth-ID. Toto Auth-ID je v podstatě bezpečnostní kód sloužící k autorizaci požadavku na převod. Vlastník domény ho získá na vyžádání u aktuálního registrátora domény. Kód služby je Domain.XXX.Transfer.Y, kde XXX se nahrazuje příslušným TLD a Y udává dobu prodloužení (tj. např. Domain.COM.Transfer.2). Struktura celého příkazu je následující:

```
<Commands>  
<Command>  
  <OrderService>  
    <!-- kod objednavkove sluzby - transfer COM domeny s prodlouzenim o 1 rok -->  
    <ServiceCode>Domain.COM.Transfer.1</ServiceCode>  
    <!-- nazev domeny pokud je nezname její ID, jinak ID -->  
    <Name>test.com</Name>  
    <!-- autorizacni kod pro zahajeni transferu -->  
    <AuthID>ABC123</AuthID>  
    <!-- Pokud True, server se bude automaticky snazit prodlouzit -->  
    <!-- domenu, pokud False, platce bude pouze upozonen na expiraci domeny -->  
    <AutoRenew>True</AutoRenew>  
  </OrderService>  
</Command>  
</Commands>
```

Prodloužení generické domény

Prodloužení doby expirace domény je placenou objednávkovou službou. Prodloužení je možné v součtu maximálně na 10 let. Prodloužení generických domén je implementováno jako reálné prodloužení domény v CR. Struktura celého příkazu je následující:

```
<Commands>  
<Command>  
  <OrderService>  
    <!-- kod objednavkove sluzby - prodlouzeni COM domeny o 2 roky -->  
    <ServiceCode>Domain.COM.Renew.2</ServiceCode>  
    <!-- nazev domeny pokud je nezname její ID, jinak ID -->
```

```
<DomainID function="1">DomainNameToID("test.com")</DomainID>
<!-- <DomainID>123</DomainID> -->
</OrderService>
</Command>
</Commands>
```

Další příkazy

Tato část obsahuje popis různých příkazů, které není možné zahrnout do žádné z předchozích kategorií.

Přihlášení uživatele

Příkaz Login slouží k přihlášení uživatele. Přihlášení zajišťuje uživateli přístup k jeho účtu, umožňuje mu volat příkazy pro práci s jeho doménami, hradit faktury z kreditu apod. Struktura příkazu je následující:

```
<Commands>
  <Command ID="0">
    <Login>
      <!-- přihlasovací jmeno -->
      <LoginName>test</LoginName>
      <!-- přihlasovací heslo -->
      <Password>test</Password>
    </Login>
  </Command>
</Commands>
```

Přílohy

Přehled nejdůležitějších dotazů pro příkaz Query

Následující tabulka uvádí přehled identifikátorů pro volání Query serveru IPAS včetně popisu parametrů, které jsou při volání dotazu třeba předat. Tabulka je platná jak pro PHP knihovnu clsIpas, tak pro .NET knihovnu Ignium Kernel. Parametr AccID je zde uváděn pouze pro úplnost, při spuštění dotazu je vždy nahrazen hodnotou ID aktuálně přihlášeného uživatele a nemá tedy smysl ho uvádět.

<i>ID</i>	<i>Parametr</i>	<i>Popis</i>
LIST_ACCOUNT_CONTACTS		Seznam kontaktů pro account
	AccID	ID accountu
LIST_ACCOUNT_SUBJECTS		ZRUŠENO
	AccID	ID accountu
LIST_ACCOUNT_DOMAINS		Seznam domén pro account
	AccID	ID accountu
LIST_ACCOUNT_VCONTACTS		Seznam virtuálních kontaktů pro account
	AccID	ID accountu
LIST_ACCOUNT_FTP_SERVERS		Seznam FTP serverů pro account
	AccID	ID accountu
LIST_ACCOUNT_EMAIL_SERVERS		Seznam mail serverů pro account
	AccID	ID accountu
LIST_ACCOUNT_WEBHOSTINGS		Seznam WWW serverů pro account
	AccID	ID accountu
LIST_ACCOUNT_DATABASES		Seznam databází pro account
	AccID	ID accountu
28		Seznam nameserverů pro account
	AccID	ID accountu
350		10 nejdříve expirujících domén pro account
	AccID	ID accountu
650		Informace o TLD dle názvu TLD
	Tld	Název TLD – např. CZ, EU, COM apod.
LIST_ACCOUNT_ORDER_INFO		Seznam objednávek u accountu
	AccID	ID accountu
LIST_ACCOUNT_ORDERS		Detailní přehled objednávek u accountu
	AccID	ID accountu
LIST_ACCOUNT_ORDERS_AUTH		Seznam objednávek čekajících na autorizaci u accountu s ID pro přeposlání autorizace
	AccID	ID accountu

<i>ID</i>	<i>Parametr</i>	<i>Popis</i>
LIST_ACCOUNT_NSSETS		Seznam NSSETů pro account
	AccID	ID accountu