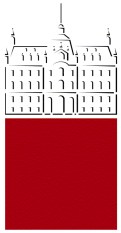
Univerza
v Ljubljani

Fakulteta
za gradbeništvo
in geodezijo



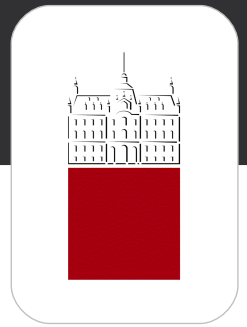
Uvod v XML

dr. Matevž Dolenc
http://kgi.fgg.uni-lj.si/pouk/racinf

Elektronski dokumenti

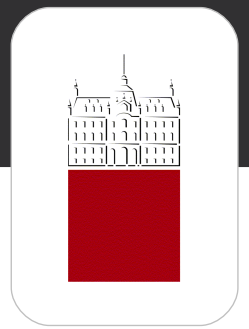
- Podatki (informacije), ki jih procesirajo računalniki so shranjeni
  - v datotekah,
  - v podatkovnih bazah (vmesnik med uporabniki in fizičnimi datotekami s podatki).
- Struktura podatkov v datoteki je praviloma odvisna od programov, ki jo procesirajo.
  - Word ima svoj obliko zapisa.
  - Excel ima svojo obliko zapisa.
  - Format podatkov v podatkovnih bazah je spet drugačen.

Procesiranje elektronskih podatkov



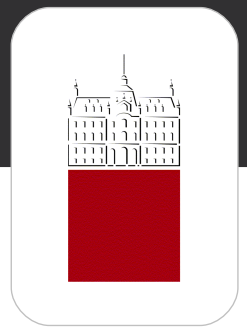
- Samostojni programi
  - Programi delujejo samostojno.
- Integrirani informacijski sistemi
  - Obravnavajo dokumente izdelane z različnimi programi.
- Porazdeljeno procesiranje
  - Procesiranje podatkov z računalniki v omrežju.
- Povezovanje različnih vrst podatkov v programih ni preprosto.
  - Vsaka datoteka ima svojo strukturo podatkov.
  - Programiranje datotek ni preprosto. Poznati je potrebno strukture podatkov za vsako vrsto dokumenta.
  - Odpravljanje skritih napak v programih je zamudno (visoke cene programov).
- Rešitev: podatkovni standard.

Zahteve po kompleksnih sistemih naraščajo



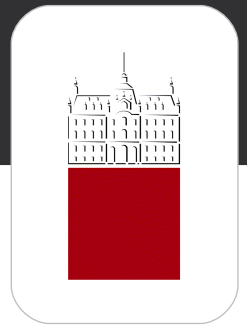
- Zmogljivost računalnikov stalno narašča.
  - Programi lahko samodejno opravijo določene naloge.
  - Sestavljanje programov iz komponent poenostavlja njihovo izdelavo.
- Izdelava kompleksnih sistemov je draga in počasna.
  - Podatki in rezultati različnih programov so zapisani vsak po svoje (Word, Excel, Access, ...).
  - Na primer, za izmenjavo inženirskih načrtov se je uveljavil dwg-format, ki pa ga uporablja program AutoCAD.

Standardni format za podatke? XML



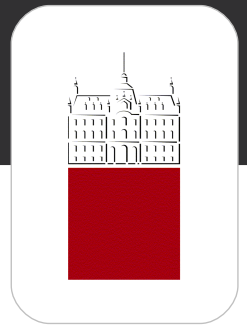
- Svetovni standard za enoten zapis elektronskih dokumentov se imenuje XML.
- Določen je bil 1998, uveljavil se je leta 2001.
- Na temelju XML so nastali še drugi standardi.
  - XSLT določa transformacijo XML-dokumentov.
  - XML-Schema določa podrobno strukturo dokumentov.
- XML-tehnologija je sklop standardov za procesiranje elektronskih dokumentov, ki upoštevajo standard XML.
  - XML je temelj.

Kaj je XML



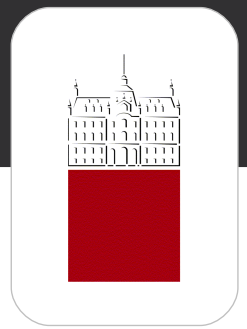
- EXtensible Markup Language
- Kratica pomeni
  - Razširljivi Označevalni Jezik.
    - Razširljiv. XML omogoča definicijo in uporabo XML-jezikov.
    - Označevalen. Podatki so označeni. Jezik HTML je primer jezika, ki uporablja označene podatke.
    - Jezik. XML določa pravila (slovnico), ki jo morajo upoštevati vsi XML-jeziki.
- XML je standard, ki
  - določa samo nekatera splošna pravila XML-jezikov.
  - vsebuje jezik za definicijo konkretnih XML-jezikov.

Spletna stran v jeziku HTML – primer označenega dokumenta



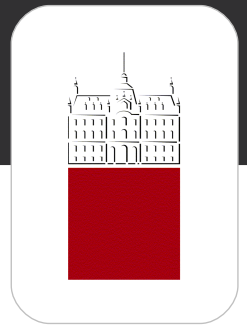
```
<html>
   <head>
     <title>Projekt ...</title>
   </head>

   <body>
     <h1>Projekt ...</h1>
     <p>
            ...
     </p>
   </body>
</html>
```

Kaj je XML



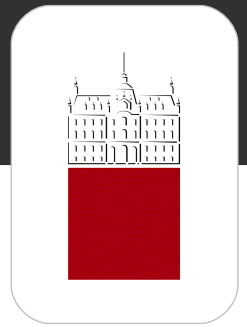
- Internetski standard
  - http://www.w3.org/TR/REC-xml
- Razvit je bil 1998 na osnovi standarda SGML.
- Standard za zapis elektronskih dokumentov (XML-dokumentov)
  - XML-dokument je besedilo.
    - Zadošča navaden urejevalnik besedil (Notepad).
  - XML-dokumenti so praviloma razumljivi računalnikom in ljudem.
    - Vsebuje drevesasto strukturo XML-elementov.
    - Vsebuje tudi procesne inštrukcije (dodatna navodila programom, kako je potrebno obdelati dokument).

Kaj XML ni



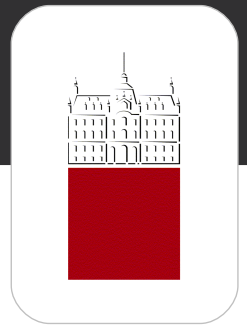
- XML ni programski jezik. Ne obstaja prevajalnik za XML.
- XML ni mrežni prenosni protokol za prenos podatkov.
  - XML ne pošilja podatkov po omrežjih.
- XML ni podatkovna baza.
  - XMl ne more nadomestiti Oracle, MySQL, Access, SQL Server, ...

Primer preprostega XML-dokumenta



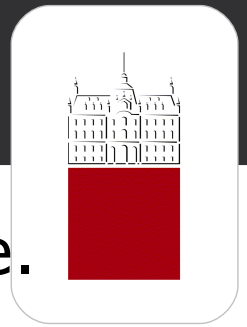
```
<?xml version="1.0 encoding="windows-1250" ?>

<e-sporočilo datum="26.5.2000">
  <pošiljatelj>Peter Novak</pošiljatelj>
  <naslovnik>Miha Kovač</naslovnik>
  <naslov-sporočila>Zaključno poročilo</naslov-sporočila>

  <vsebina>
    Do četrtka pošlji ....
  </vsebina>
</e-sporočilo>
```

Zapis besedil z računalniki



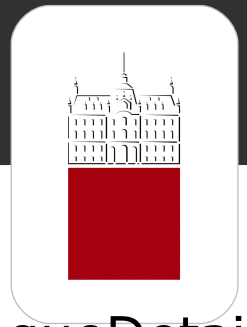
- Vsak znak je kodiran.
  - Koda ali šifra je niz bitov.
  - ASCII koda: 1 znak 7 bitov, 128 znakov.
  - ANSI koda: 1 znak = 1 bajt (8 bitov), 256 znakov.
- Operacijski sistem Windows uporablja kodne strani.
  - Kodna stran dooloča kodiranje znakov po sistemu ANSI
    - 1 bajt = 1 znak
  - Obstaja veliko kodnih strani, ki jih lahko namestimo.
- Po svetu se uporablja mnogo znakov: cca. 50000.

Kodna stran 1250



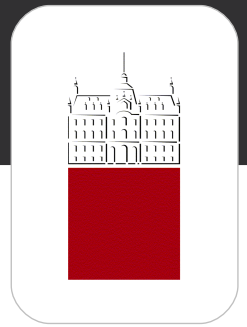
- Primerno za nekatere vzhodno evropske države.
- Slovenska Okna uporabljajo 1250.
  - windows-1250
- Tipkovnice so prilagojene našim znakov.
  - http://www.microsoft.com/truetype/unicode/1250.htm

Univerzalni sistem znakov



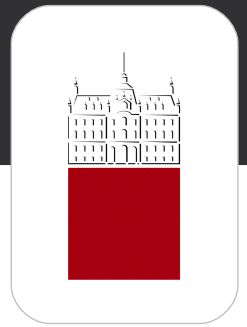
- ISO standard: ISO-IEC 10646
  - http://www.iso.ch/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=39921
  - ISO standarde moramo plačati!
- Enakovreden (skoraj) je javno dostopen standard Unicode
  - ISO-10646-UCS-2
  - http://www.unicode.org/standard/standard.html
  - http://www.unicode.org/versions/Unicode4.0.0/
  - http://www.alanwood.net/unicode/

Lastnosti sistema Unicode



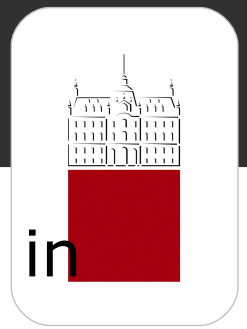
- Vsakemu znaku določa ime in zaporedno številko.
- Število znakov cca. 50000
- Ne določa grafične podobe znaka (fonti).
- Ne določa podrobno, kako kodiramo znake.
  - 1 znak = 1 bajt, 1 znak = 2 bajta
- Znakom določa numerične reference:
  - &#številka-znaka;
  - Ɵ
  - &#xšestnajstiška-številka-znaka;
  - epsilon: ε

Kodirni sistemi za Unicode



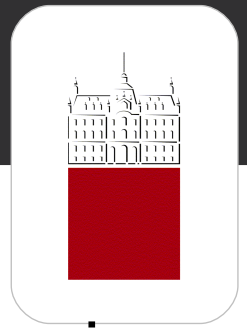
- UTF-8
  - Znaki so kodirani z različnim številom bajtov.
    - ASCII znaki uporabljajo 1 bajt.
    - Ostali znaki se kodirajo v 2 ali 3 bajte.
  - Je primeren sistem za Slovenijo.
  - Standard XML privzeto uporablja UTF-8.
  - http://www.cl.cam.ac.uk/~mgk25/unicode.html#utf-8
- UTF-16
  - Znaki so kodirani z 2 bajtoma. Datoteke so daljše.
  - Primerno za delovni zapis znakov v programih.

Zgradba XML-dokumenta



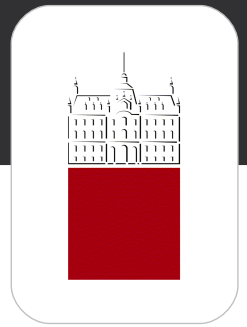
- Dokument vsebuje prolog, procesne inštrukcije in drevesasto strukturo XML-elementov.
- Prolog vsebuje XML-deklaracijo in procesne inštrukcije.
- XML-element vsebuje podatke dokumenta.
  - Ima predpisano strukturo.
  - Dokument ima en sam izhodiščni XML-element.
- Procesne inštrukcije so dodatna navodila programom, ki obdelujejo XML-dokumente.

XML deklaracija



- XML deklaracija ni obvezna.
- Izgleda kot procesna inštrukcija; tehnično pa to ni.
- Primer deklaracije:
  - `<?xml version="1.0" encoding="windows-1250" ?>`
  - Do sedaj obstaja samo verzija 1.0.
  - Verjetno bodo v prihodnosti nastale nove verzije.
- XML-dokument je besedilo, niz znakov.
  - Znaki so v računalniku vedno kodirani (šifrirani).
- Atribut encoding določa kodirni sistem znakov.
  - V uporabi je mnogo kodirnih sistemov.
  - Privzet kodirni sistem je UTF-8.

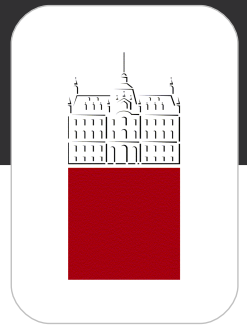
XML-element: drevesasta struktura XML-elementov



- XML-dokument vsebuje drevesasto strukturo XML-elementov.
  - XML element ima začetno oznako, vsebino in končno oznako.
  - `<ime>`vsebina ...`</ime>`
  - Oznaka vsebuje ime v zašiljenih oklepajih.
- Vsak dokument ima en sam začetni oz. izhodiščni (korenski) element.
- Imenujemo ga tudi dokumentni element.

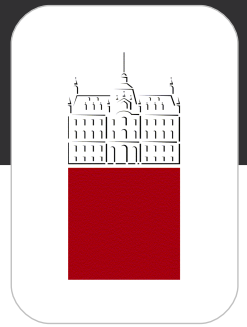
```
<dokument>
...
</dokument>
```

XML-element: drevesasta struktura XML-elementov



- Oznake nastopajo XML-elementu v paru začetna in končna oznaka (ang. tag).
  - `<ime-elementa>` določa začetek xml-elementa.
  - `</ime-elementa>` določa konec xml-elementa.

Vsebina XML-elementa sme biti



- Ničesar, presledki, beli znaki (tab, znak za novo vrsto).

```
<v></v>
```

```
<v>
</v>
```

```
<v>    </v>
```

- Podatki smejo biti v atributih v začetni oznaki.

```
<točka x="10" y="5"/></točka>
```

- Besedilo niz znakov, ki ne vsebuje znaka < ali &,

```
<v>To je besedilo</v>
```

- En ali več xml-elementov.

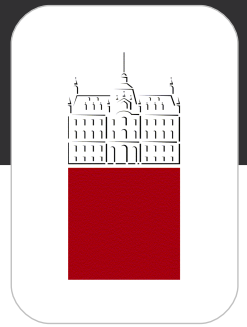
```
<v><ime>Peter</ime></v>
```

- Mešanica besedila in xml-elementov.

```
<v><ime>Peter</ime><priimek>Grilc</priimek></v>
```

```
<v>Podpisani <ime>Peter</ime><priimek>Grilc</priimek> bom ...</v>
```

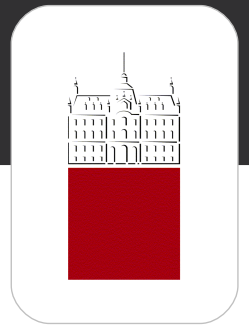
XML-elementi brez vsebine



- Dovoljen je xml-element, ki nima vsebine.
  - Ima samo eno oznako (začetno), ki je tudi končna.
  - Oznaka se konča z `/>`.
  - Primer: `<element-brez-vsebine a="1" b=0"/>`
  - V oznaki smejo biti atributi, ki so razloženi v nadaljevanju.
  - Končne oznake ni.
- Primer:

```
<velikost d="10" h="20"></velikost>
<velikost d="10" h="20“/>
```

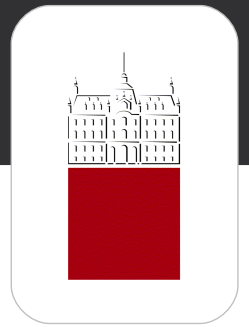
XML-element, ki vsebuje besedilo



- Primer:

```
<Dolžina>10</Dolžina>
<Opis>
Parcela se nahaja na težko dostopnem področju.
Dostop z avtom ni možen.
</Opis>
<Pravilo>x &lt; 10</Pravilo>
```

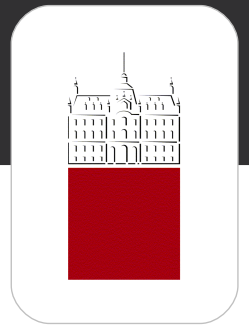
XML element z mešano vsebino



- Vsebina je besedilo pomešano z xml-elementi.

```
<Opis>
Parcela se nahaja na <b>težko</b> dostopnem področju.
  <i>Dostop z avtom ni možen.</i>
</Opis>
```

Pravila za pisanje xml-elementov



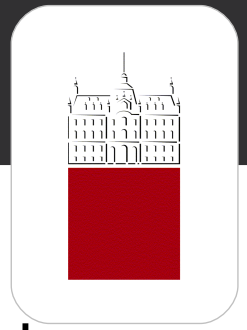
- Gnezdenje XML-elementov je dovoljeno

```
<parcela>
  <lastniki>
    <lastnik><ime>Peter</ime><priimek>Kovač</priimek></lastnik>
    <lastnik><ime>Miha</ime><priimek>Trata</priimek></lastnik>
  </lastniki>
</parcela>
```

- Križanje xml-elementov je prepovedano.

```
<parcela>
  <lastniki>
    <lastnik><ime>Peter</ime><priimek>Kovač</priimek></lastnik>
    <lastnik><ime>Miha</ime><priimek>Trata</priimek></lastnik>
  </parcela>
</lastniki>
```

Atributi



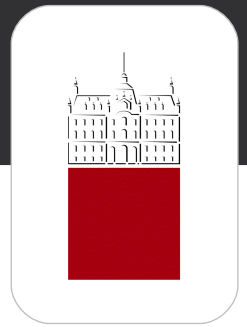
- Atributi vsebujejo podatke xml-dokumenta.
- Začetna oznaka XML-elementa sme imeti atribute.
- Primer xml-elementa z atributi:

```
<izpit datum="12.6.2000" id='215348'>
      ...
</izpit>
```

- Oblika atributa:
  - `ime = "vrednost"`
  - `ime = 'vrednost'`
  - isto ime je lahko v oznaki samo enkrat
- Oba zapisa sta enakovredna. Dovoljena znaka za narekovaj sta " in '.

```
<avtor priimek="Al'Capone"> … </avtor>
```

Atributi



- Končna oznaka ne sme imeti atributov.
- Število atributov ni posebej omejeno.
- V oznaki ne sme biti dveh atributov z istim imenom.

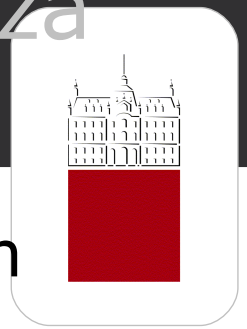
```
<točka x="3" x="2">
```

- Vrstni red atributov v oznaki ni pomemben.

```
<točka y = "2"    x = "3">
```

- Pred in za = sme biti poljubno število belih znakov.

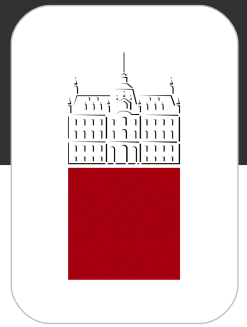
Zapis znakov, ki se sicer uporabljajo za zapis xml-oznak in atributov



- XML pozna naslednje simbole, ki imajo poseben pomen.
  - `<` (pomeni znak `<`)
  - `>` (pomeni znak `>`)
  - `&` (pomeni znak `&`)
  - `&quot;` (pomeni znak `"`)
  - `&apos;` (pomeni znak `'`)
- Primer:
  - `x > 10` in `x < 20` smemo zapisati z:

```
<izraz>x &gt; 10 in x &lt; 20</izraz>
```

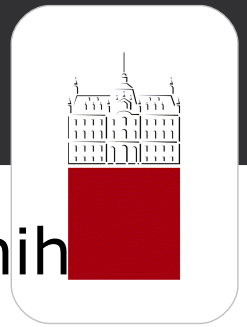
CDATA sekcija: <![CDATA[...]]>



- Omogoča, da podatki vsebujejo znake, ki imajo sicer v dokumentu poseben pomen:
  - < moramo pisati s simbolom <
  - & moramo pisati s simbolom &
- Primer:

```
<![CDATA[
    <svg width="12" height="10">
   <ellipse rx="100" ry="140"/>
      <rect x="6" y="2" width="8" height="5"/>
   </svg>
]]>
```

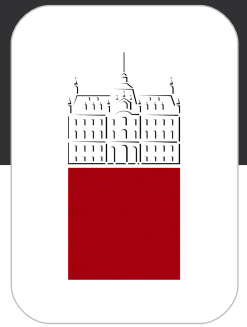
Komentarji v XML-dokumentu



- Namen komentarjev je komentiranje posameznih delov dokumenta.
- Komentar ne spada med podatke dokumenta.
- Navodilo, opazka avtorja dokumenta drugim avtorjem.
- Primer:

```
<!-- To je komentar -->
<!-- Tudi
        to
        je
        komentar -->
```

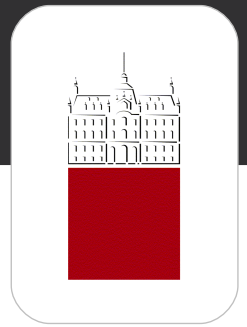
Uvod v imenske prostore



- V XML-dokumentu ima lahko ista oznaka več pomenov.
  - Element `<naslov>` ima lahko več pomenov. Naslov dokumenta, naslov poglavja, naslov slike.
  - `<naslov> ...</naslov>`
- Imenski prostori omogočajo razlikovanje imen.

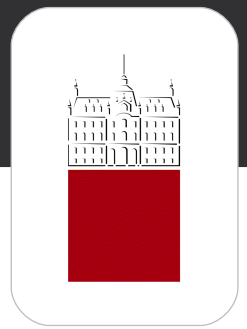
```
<kt:dokument xmlns:kt="ko.fgg.uni-lj.si">
    <kt:naslov>To je naslov ... </kt:naslov>
    ...
</kt:dokument>
```

Imenski prostori in XML



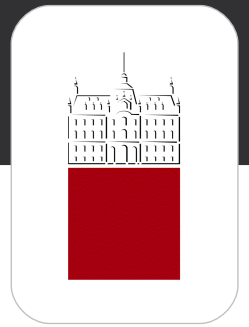
- Imenski prostor je zbirka enoličnih imen.
  - Imenski prostor ima svoje ime, oznako.
- Določa ga URI:
  - Universal Resourse Identificator (URI)
  - Obstaja standard za strukturo URI.
  - Pogosto je URI podoben URL (Universal Resourse Locator), vendar ne določa datoteke, ampak globalno enolično oznako (enolično ime).
- Primeri:
  - fgg.uni-lj.si
  - kgi.fgg.uni-lj.si

Uporaba imenskih prostorov v XML-dokumentih



- Imenskemu prostoru v dokumentu priredimo (kratko) predpono, ki nadomesti (dolgo) ime imenskega prostora.
  - <predpona:ime-elementa>
- V začetni oznaki xml-elementa uporabimo poseben atribut xmlns, ki imenskemu prostoru priredi predpono.
  - xmlns:predpona="ime imenskega prostora - URI"
- Predpona je običajno kratek niz znakov (dolg le nekaj znakov).
- Ime imenskega prostora praviloma ni kratko (enolično).
- Uporabljamo lahko več imenskih prostorov.

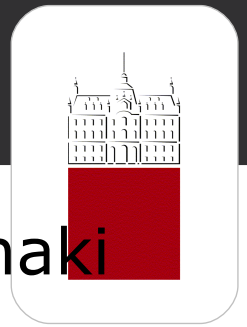
Uporaba imenskih prostorov v XML-dokumentih



- Primer:

```
<kt:kataster xmlns:kt="kataster.fgg.uni-lj.si"
                        xmlns:rs="predpis.rs.si">
    <kt:ko>Katastrska občina ... </kt:ko>
    ...
    <rs:dohodek> ...</rs:dohodek>
</kt:kataster>
```

Veljavnost imenskega prostora.

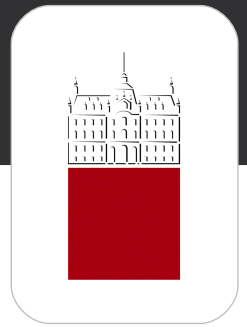


- Imenski prostor vedno definiramo v začetni oznaki določenega xml-elementa.
  - Napišemo atribut xmlns.
  - Velja v celem elementu, tudi v oznaki tega elementa!
  - Primer:

```
<p:daljica xmlns:p="http://geox.kranj.si">
   <p:t1 x="5" y=4"/>
   <p:t2 x="15" y=-4"/>
</p:daljica>
```

- Če ga določimo v začetnem (izhodiščnem) elementu, velja v celotnem xml-dokumentu.

Privzeti imenski prostori

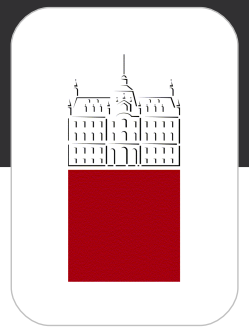


- Dovoljeno je, da predpone ne definiramo.
  - `xmlns` brez predpone.
  - Imenujemo ga privzeti imenski prostor.
  - Primer:

```
<svg xmlns="http://www.w3.org/2000/svg"
   width="12cm" height="10cm">
   <ellipse rx="100" ry="140"/>
   <rect x="6cm" y="2cm" width="8cm" height="5cm"/>
</svg>
```

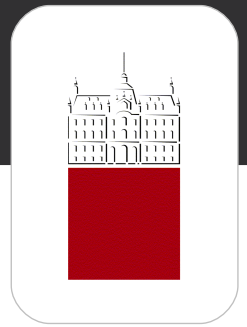
- Element `svg` in vsi vsebovani elementi pripadajo imenskemu prostoru, ki ga določa atribut `xmlns`.
- Atributi ne pripadajo privzetemu imenskemu prostoru. Ne pripadajo nobenemu prostoru.

Pravilno oblikovani xml-dokumenti (well formed)



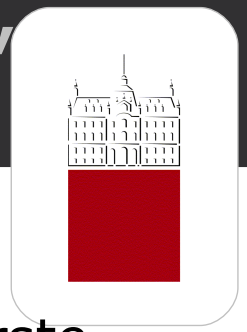
- Na začetku sme biti posebna procesna inštrukcija, ki določa začetek xml-dokumenta:
  - `<?xml version="1.0"?>`
- Posebej lahko določimo kodirni sistem znakov. Privzeto velja Unicode kodirni sistem UTF-8.
- Slovenska okna uporabljajo kodne strani:
  - `<?xml version="1.0" encoding="windows-1250"?>`
- Upoštevajo splošna pravila za pisanje XML-elementov.
- Dokument ima samo en izhodiščni (dokumentni) element.
- Elementi se ne smejo križati. Gnezdenje je obvezno.
- Za nekatere znake moramo uporabiti posebne simbole: (`> &apos;`).

Določitev jezika v xml-dokumentu



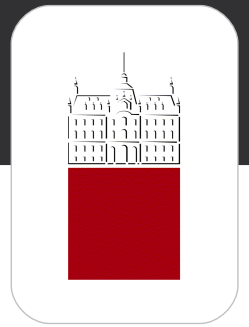
- V začetni oznaki xml-elementa smemo napisati poseben atribut, ki določa jezik:
- `xml:lang="koda-jezika"`
  - Vsi jeziki imajo standardne oznake (kode).
  - `xml:lang="sl"`
  - `<predpis xml:lang="sl"> ... </predpis>`
- Atribut velja samo znotraj elementa.
- Atribut, ki določa jezik lahko uporabijo programi, ki procesirajo xml-dokument.

Kako definiramo nov XML-jezik, jezik, ki ustreza standardu XML?



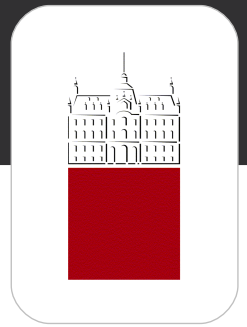
- Napišemo slovnico za tak jezik.
  - Slovnica določa pravila za zapis podatkov določene vrste (dokumentov).
  - Slovnico zapišemo v svoji datoteki (pripona DTD).
- Slovnico imenujemo tudi:
  - Document Type Definition (DTD).
  - Definicija tipa dokumenta.
- Za zapis slovnice uporabimo jezik DTD, ki ga določa standard XML.
  - Slovnica določa dovoljene elemente.
  - Vsakemu elementu se določi: ime, atributi in vrsta vsebine.
  - Slovnica določa tudi strukturo xml-elementov.

Primer slovnice DTD



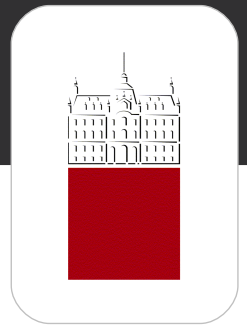
```
<?xml version="1.0 encoding="windows-1250" ?>

<!ELEMENT sporočilo
        (pošiljatelj, naslovnik, vsebina)>

<!ELEMENT pošiljatelj (#PCDATA)>
<!ELEMENT naslovnik (#PCDATA)>
<!ELEMENT vsebina (#PCDATA)>

<!ATTLIST sporočilo  datum  CDATA #REQUIRED>
```

Veljavni xml-dokumenti (valid)



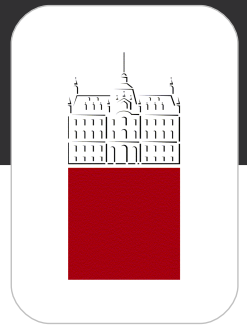
- Upoštevajo slovnico jezika xml-dokumenta.
  - Document Type Definition določa slovnico.
  - DTD-datoteka.
- Vsebujejo samo oznake in atribute, ki so določeni v slovnici jezika, ki mu pripada xml-dokument.
  - Slovnica določa tudi vrstni red vsebovanih elementov.
- Veljavnost xml-dokumentov lahko preverjajo programi, ki procesirajo xml-dokumente.
  - Ob napaki lahko javijo sporočilo, ki določa tudi mesto napake v dokumentu.
  - Kontrola veljavnosti je lahko procesno zamudna.
  - Veljavnosti zato ne kontroliramo pri vsakem dokumentu.

XML dokumenti brez definirane slovnice



- Uporabljati smemo tudi XML-dokumente brez definirane slovnice.
  - Veljavnosti dokumentov tedaj ne moremo preveriti.
  - Take dokumente pogosto uporabljamo.
- Jezik HTML je primer označevalnega jezika:
  - Upošteva starejši standard SGML, ki tudi pozna koncept slovnice jezika.
  - Večina spletnih strani se ne sklicuje na slovnico, ki naj bi jo upoštevala. Kontrole ni.
  - Brskalniki procesirajo tudi napačno zapisane spletne strani. Predstavijo stran, kar najbolje se da.

Atributi ali elementi?

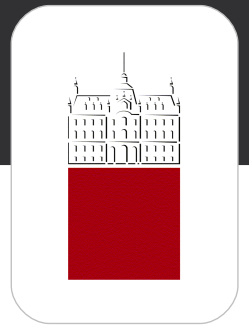


- Podatke dokumenta lahko določajo atributi ali elementi.

```
<lega x="10" y="20">
<lega>
   <x>10</x>
   <y>20</y>
 </lega>
```

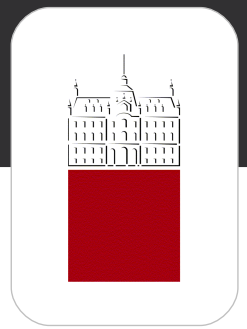
- Pravila, kdaj je bolje nek podatek predstaviti kot element ali kot atribut ni.
  - Vsebina atributa je atomarni podatek.
  - Zapis atributa je krajši.
  - Vsebina elementa je lahko sestavljena iz elementov.
  - Elementi so bolj prožni, saj lahko kasneje znotraj elementa dodamo še kak nov element.

Primer dokumenta za zapis podatkov parcele



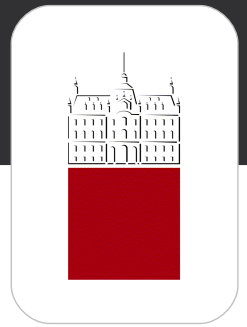
```
<parcela id="P32/2">
   <vrsta-rabe>8</vrsta-rabe>
       <centroid x="23.122" y="18.322"/>
       <poligon>
            <v x="1.54" y="2.10"/>
            <v x="9.53" y="7.28"/>
            ...
       <poligon>
</parcela>
```

Uporaba XML standarda



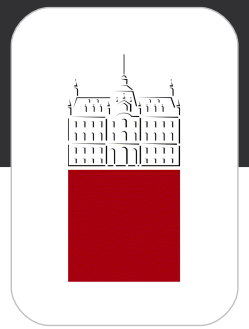
- XML lahko uporabimo za zapis poljubnih dokumentov.
- XML-dokument je razumljiv ljudem in računalnikom.
- XML je prenosni format podatkov med programi.
- Najbolj je namenjen programerjem, da hitreje in bolj zanesljivo izdelajo programsko opremo.
- XML je besedilo, zato ga lahko urejamo z navadnim besedilnim urejevalnikom.
- Za procesiranje XML dokumentov v programih uporabljamo programska orodja (komponente).
  - Taka orodja so že del razvojnih sistemov (Visual Studio.Net)
- Večina sodobnih Internet brskalnikov ima vgrajen xml-procesor.

Zakaj se je XML pojavil tako pozno?



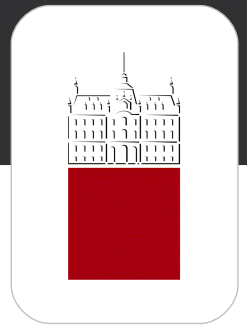
- XML dokument je besedilo - niz kodiranih znakov..
  - Procesiranje podatkov predstavljeno z znaki, je lahko počasnejše od binarne predstavitve.
  - Binarni podatki so hitrejši.
- Zapis XML dokumenta je daljši od zapisa podatkov brez oznak.
  - Diski so bili včasih majhni.
  - Procesorji so bili počasni.
  - Danes so računalniki zelo zmogljivi, tudi osebni.
- Pri uporabi dokumentov pogosto potrebujemo podatke (numerične) v binarni obliki.
  - Potrebne so pogoste transformacije iz besedila v števila.

Kako programiramo XML-dokumente?



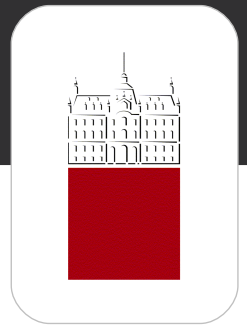
- Za programiranje XML-dokumentov imamo na razpolago že izdelane programske gradnike.
  - Tak gradnik ima lahko metodo `load`, ki iz datoteke prebere celoten XML-dokument in ga zapiše v pomnilnik.
  - Gradnik ima metode za doseganje poljubnih elementov in njihovih atributov.
  - Gradnik ima tudi metode za urejanje elementov in njihovih atributov.
  - Gradnik ima tudi metodo `save`, ki omogoča shranjevanje programsko urejenih XML-dokumentov.
- Gradnik lahko uporabimo v poljubnem programskem jeziku (VB.Net, C#, C++, Java, Python, ...)

XML-procesorji



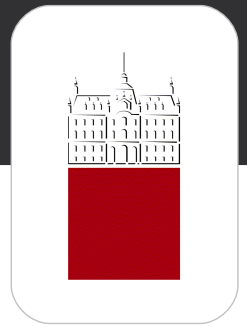
- XML- razčlenjevalniki (parserji, procesorji) so programska orodja, ki omogočajo izdelavo programov, ki procesirajo xml-dokumente.
  - Podatki xml-dokumentov se predvsem uporabljajo v programih!
  - Programi procesirajo podatke.
- Večina današnjih programskih jezikov ima že narejena taka orodja (komponente)
  - Java
  - Visual Studio.Net
  - Sklop komponent za procesiranje xml-dokumentov.

Document Object Model - DOM



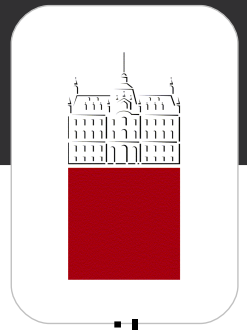
- Programski predmeti za modeliranje XML dokumentov so standardizirani.
  - Metode in lastnosti programskih predmetov so določene.
- DOM je abstraktni predmetni model, ki določa programske predmete za predstavitev in procesiranje XML-dokumentov.
  - DOM je specifikacija, standard.
  - Standard, ki določa izdelavo konkretnih procesorjev xml-dokumentov v različnih programskih jezikih, ki podpirajo predmetni pristop.
  - Standard je neodvisen od programskega jezika.

XHTML



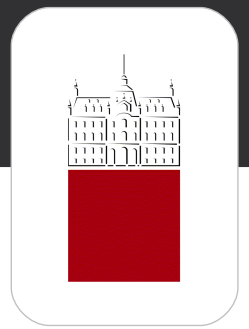
- EXtensible Hypertext Markup Language
- XMTML je različica jezika HTML, ki upošteva splošna pravila XML-dokumentov.
- Upoštevajo ga vsi pomembni spletni pregledovalniki.
  - Znajo predstaviti stran zapisano po pravilih XHTML.
- XHTML je obenem tudi prečiščen HTML, ki nima nekaterih elementov, ki se niso uveljavili.
- XHTML naj bi postopoma v spletnih straneh nadomestil HTML 4.0.
  - HTML 4.0 bo še vedno v uporabi!

XHTML: lastnosti



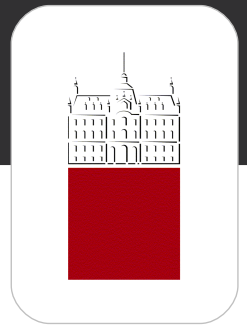
- XHTML se skoraj ujema z HTML 4.0.
- XHTML je bolj strog. Upošteva XML-pravila za pravilno oblikovanje xml-dokumentov.
- Vse oznake vsebujejo samo male črke.
- Vsi elementi morajo biti zaključeni!
  - `<br/>` in ne `<br>`
  - `<p>...</p>` in ne samo `<p>`
- Atribute moramo obvezno pisati v narekovajih (enojnih ali dvojnih).
  - Narekovajev ne smemo izpuščati.
- Atribut `id` lahko identificira element.

Primeri že definiranih XML-jezikov



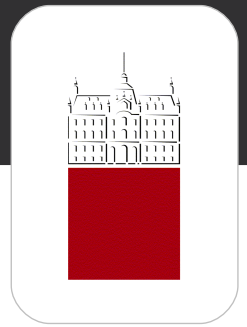
- MathML: jezik za pisanje matematičnih izrazov.
- SVG (Scalable Vector Graphics): jezik za zapis vektorskih grafičnih elementov (slik).
  - Vsi vektorski grafični programi bodo v kmalu podpirali SVG.
  - SVG procesor lahko dodamo v spletni pregledovalnik.
- e-Book: jezik za elektronske knjige.
- SOAP: jezik za zapis programskih predmetov.
- SMIL: Synchronized Mutimedia Integration Language
- ...

XML v gradbeništvu



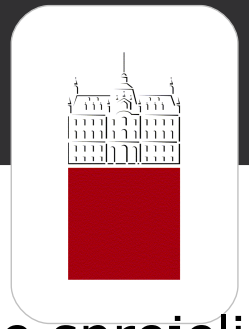
- Produktni informacijski model
  - IFC (Industry Foundation Class)
    - http://www.iai-international.org/Model/IFC(ifcXML)Specs.html
  - zapis IFC modela
    - STEP physical file, ISO 10303-21
    - xmlIFC
- ...

XML v geodeziji



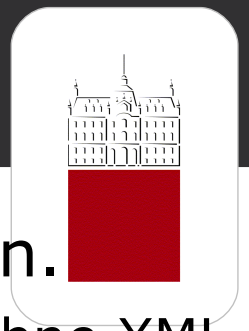
- GML: Geography Markup Language
- Jeziki za uradne dokumente pri uporabi katastra, zemljiške knjige, ...
- Jeziki za zapis geodetskih standardov
  - Današnje standarde zelo težko računalniško procesiramo. Prirejeni so samo za ljudi.
- Jeziki za predstavitev geografskih informacijskih sistemov.

Druga uporaba XML



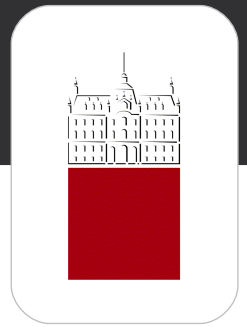
- XML zelo vpliva na elektronsko poslovanje.
  - Vsi pomembnejši proizvajalci poslovnih sistemov so že sprejeli XML.
  - Komunikacija poteka z xml-dokumenti.
- Izdelava poslovnih spletnih rešitev se je bistveno poenostavila.
  - Bančna transakcija je XML-dokument.

Slabe lastnosti XML-standarda



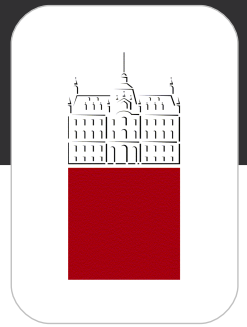
- Zapis XML-dokumentov je praviloma dolgovezen.
  - Za ročno pisanje XML-dokumentov potrebujemo posebne XML-urejevalnike.
  - Procesiranje XML-dokumentov je lahko bolj počasno kot procesiranje datotek, ki vsebujejo samo podatke.
  - Program AutoCAD ne podpira zapisa slik po standardu XML.
- XML standard je sicer kratek, vendar dovolj zapleten.
- Zapis slovnice XML-jezika z DTD ni XML-dokument.
  - Obstajajo alternativni standardi za zapis slovnic.
  - XML-Schema. Vsebuje tudi numerične in druge tipe podatkov.

Drugi standardi povezani z XML



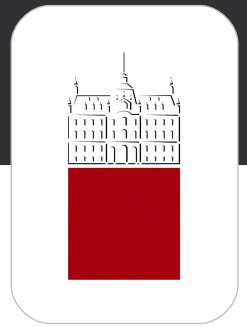
- XML-Path določa sklicavanje na posamezne dele XML dokumentov.
  - Zelo važen standard.
  - Podoben je SQL jeziku pri relacijskih podatkovnih bazah.
- XML-Pointer določa posplošene hipertekstne povezave.
  - Lahko vsebuje tudi reference na dele XML-dokumentov.
  - Standard v razvoju.
- XQuery je pomemben standard za poizvedovanje vsebine XML-dokumentov.

Drugi standardi povezani z XML



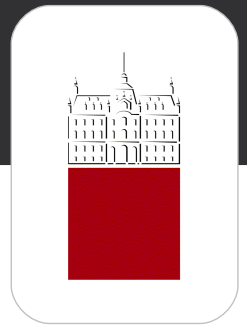
- XSLT – Extensible Style Language (Transformation)
  - Določa preoblikovanja (transformacije) XML-dokumentov.
- XSLT procesorji razumejo XSLT jezik, s katerim lahko z enostavnimi ukazi transformiramo XML-dokument ali samo del dokumenta v naslednje oblike:
  - drug XML-dokument
  - v HTML
  - v navadno besedilo.
- XSLT transformacija je zapis (neke vrste program) v XML-jeziku XSLT.

XML-Schema



- XML.Schema ima naslednje lastnosti
  - Je boljši od jezika DTD, ki je vgrajen v standard XML.
  - Obsežen in razmeroma zapleten standard.
  - Temelj za razvoj novih orodij, ki temeljijo na standardu XML.
  - Zapis slovnice z XML-Schema je XML-dokument.
- Omogoča precizno določanje sintakse večine današnjih dokumentov.
  - Zelo podrobno se ukvarja s podatkovnimi tipi.
  - Standardno vključuje večino podatkovnih tipov, ki se uporabljajo v današnjih programih.
  - Omogoča definicijo izpeljanih podatkovnih tipov.

XML



- je standard, ki ima izreden vpliv na razvoj računalništva in informatike.
  - Pospešuje uporabo računalnikov (izdelava programov je cenejša)
  - Računalniki bodo samostojno opravljali naloge, pri katerih morajo danes sodelovati ljudje (računalniški agenti).
  - Omogoča, da lahko podatke predstavimo na različne načine.
- Mnogi obstoječi programi imajo svoj način za zapis podatkov in rezultatov.
  - Ti zapisi med seboj niso kompatibilni.
  - Procesiranje (programiranje) takih dokumentov zahteva preveč naporov.
  - Izdelava integriranih sistemov ni enostavna.