Prezentace 1

čtvrtek 19. ledna 2023 16:17

- necessity of correct data representation & following specifications
- data should be app-independent - multiple apps may need to share
- conceptual model → logical model → physical level
  (problem structure) (structure in given technology) (streaming files)
- conceptual model - defines entities
  - defines properties
  - defines their relations
  - UML diagrams:


entity 1 is associated with 0 - 1 of entity 2

- terms of use specify author, allowed use-cases of data, what kind of data...
- data service - data accessible via API
- data models - logical view of data - graphs - RDF - Resource description framework
  - LPG - Labeled property graph
  - hierarchies - DOM - Document Object Model
    - JSON - for both format and model
  - tables - relational model
- data formats - physical view - graphs - RDF
  - property graphs - CSV, JSON, GraphML, Cypher Scripts
  - hierarchical - DOM, JSON (XML, HTML)
  - relational - CSV, SQL
  - ways of serialization into files
- data schemas - annotations and constraints specifying how instances of data are defined
- open data formats - freely available specifications
- closed formats - limited license
  - claiming compatibility demands certification
- machine-readability - ease of processing by appropriate apps
- binary files - structure defined bit by bit
  - hex editors
- text files - uses encoding
  - text editors
  - ASCII - 7 bits = char
  - UTF-8 - 1-4 bytes = char
  - BOM - byte order mark
    - encoding type and byte order at the start
  - iso-8859-2, windows-1250
- 1 specification > multiple - producers adjust their data publication

- consumer learns 1 spec to process all data

- IETF - Internet Engineering Task Force
  - manages standards across borders
- W3C - standards for WWW - drafts (WD) → candidate (CR) → proposed (PR) → W3C (REC)
- ICANN - IP management
- MIME-type - ~~xxx~~ internet mail types
- ECMA - JSON, C# and stuff
- IDs - URI - uniform resource identifier
  - URN - name
  - URL - locator
  - IRI - Internationalized

- graph entities are identified with IRI - shortened by prefixes
- RDF - subject - predikate - object triplets
  - subjects and objects can be literals or resources
  - @prefix name of prefix: IRI
  - prefix: resource to append IRI with
  - classes used as resource types are also resources
  - blank node - nespojený s IRI
    - má node ID validní pouze v dokumentu kde se vyskytl
  - N-triplets - easy to parse
    - easily compressible
    - easily paged
    - use of prefixes shortens and makes more readable
    - ';' means the subject is the same for the next triplet
    - ',' means only the object changes
    - empty prefix names can be used
    - used prefixes don't need suffixes to be used
    - relative IRI - @base = some IRI
      - <# or / path> is then automatically prefixed
    - """ is multiline string
    - rdf:type shortens is 'a'
    - statemens can be reified into _:triple a rdf:Statement; rdf:subject atd...
    - RDF model - named graphs that are resources
      - ~~The~~ statements and facts are than made above these graphs
    - RDF datasets - set of named graphs
      - one default graph
  - N-quads - name of the graph is behind each triplet
  - Vocabulary - collection of class and property definitions with their IRIs
  - RDFS - class and class hierarchy definitions (inheritance)
  - RDF properties can exist independently of object/subject (unlike OOP)
  - domain - subjects on which property may be used
  - range - possible values of property
  - all props should have comment and label
  - RDF:List is used to specify order of classes (RDF is otherwise unordered) - (:a :b :c) shortens
    - linked list closed by rdf:nil
  - RDF:Bag - set that is

- linked lists closed by self: nil
- RDF: baby-set that is modifiable
  - open

- Open World Assumption (OWA) – truth of a statement is independent of what is known
- Closed assumption assumes no unless specifically known otherwise
- non-linked data can create ambiguity about possible data duplicity
  - ambiguity of what property names mean
  - no context and poor search for data

- WWW – 1. HTML publishing of documents
  2. URL as global identifiers
  3. HTTP for documents lookup
  4. documents connected by hyperlinks
  (also Web of documents)
  - scraping machine-readable data via 'mash-up' apps (data is hidden non-uniformly)

- Linked data – 1. URI as names
  2. HTTP for navigation
  3. provide standardized info upon lookup
  4. Links to related data
  - also the Web of Data
  - data is linked regardless of publisher (not confined by webpages)
  - RDF as data publishing
  - URL as global ID of entities (not documents)
  - HTTP accessed as data about entities
  - links to entities (not documents)
  - vocabularies - standards

Prezentace 3

čtvrtek 19. ledna 2023 21:06

- SPARQL - query for RDF
  - queries any parts of a triplet by ?var
  - first tries to match the graph pattern
  - it then plugs possible values of variables found in matched graph patterns (binding)
  - result is table of solutions
  - OPTIONAL { optional pattern } (if no corresponding pattern, the resulting table will contain NOT BOUND for vars inside)
    - can be used multiple times
  - UNION - basically logical or (returns more patterns if more match)
  - PREFIX - defines a specific
  - SELECT ?vars WHERE { triplets with ?vars }
  - BASE - for relative URIs
  - blank nodes - within document scope or result scope
    - result and document nodes may not correspond in identifiers
  - DISTINCT
  - GRAPH - allows us to name graphs
  - FILTER (condition) - within WHERE
    - bound(?v), isIri(?v), isBlank(?v), isLiteral(?v), str(?v), lang(?v), datatype(?v) - operations
  - LIMIT 100...
  - predicate/ predicate/ predicate subject - tracing a specific path to subject
  - MAX, AS
  - other SQL aggregates (MIN...)
  - BIND (expression AS ?var)
  - IF (condition, true, false) AS ?var (conditional var binding)
    - return if true
  - literals and IRIs can be constructed using constructor functions (turning vars to typed literals)
  - supports timedate data types
  - inner queries are evaluated first
  - VALUES ?var { val list } (tries to match all graph patterns with ?var on these vals)
  - ASK checks if result exists, SELECT finds it
  - CONSTRUCT formats results
  - ORDER BY at the end of queries

- OFFSET, HAVING

Prezentace 4

pátek 20. ledna 2023 16:26

- dcTerms: Dublin Core Metadata Terms – URI, Label, Definition, Type of Term, Range Includes, Subproperty of
- skos: Simple Knowledge Organization System – Concepts, ConceptSchemas (Concept Aggregates), Lexical Labels, notations, Semantic relations, Collections, Mappings
- gr: GoodRelations – Agents – Promise principle
  - Agent – an entity (Business)
  - Object or service – product or service
  - Promise – the promised outcome/activity
  - Location – where this is available
- Schema.org and GR allow publishers to publish low-quality rough data so that they can be published quickly
- Schema.org – for extraction of data from web pages
  - based on: HTML microdata, RDF, JSON-LD
- Wikidata – like wikipedia for facts, not docs
  - anyone can contribute
  - SPARQL queriable
  - run by Wikibase SW
  - Items – basic wikidata elements
    - QID – q is thing number
    - labels, descriptions, aliases
  - Statements – about items
    - PID (p for property)
    - Value
    - Qualifiers – prop, value – further specify relation
    - References – prop, value
    - Rank

Prezentace 5

pátek 20. ledna 2023 17:34

- Graphs beat relational models if large number of tables are needed
- graph use-cases – important connections between entities
  - self-referencing hierarchies
  - unbounded hierarchies (like changing networks)
  - alternative path discoveries
- Labeled prop Graph (LPG) – Nodes can have multiple labels
  - each node has its own prop set
  - labels are like types
  - oriented multigraphs
  - edges have labels too
  - nodes and edges have key-value
- Cypher – LPG querying language
  - used by Neo4j
  - LPG typically lacks standardization
  - `-->[f:Follows]->` – relationship representation
  - `(p:Person:Mamal)` – : types something (p = var)
  - `(p:Person {name:'Smith'})` – node with props
  - find patterns with MATCH pattern RETURN parts of pattern
  - MATCH can further be filtered by WHERE
  - SET can add props to vars before return
  - CREATE CONSTRAINT ON (pattern) ASSERT condition
  - MERGE finds or creates pattern
  - ON CREATE
  - returning the entire path may not match the directions of relationships
  - shortestPath(pattern)
  - only Cypher keywords are case insensitive
  - typical aggregate functions
  - missing values in results are null
  - regex matching using =~
  - WITH – manipulates pattern before continuing the query
  - loading CSV – imports the CSV → make sure the format is OK → maybe change the default string datatype
  - Neo4j Graph Data Science Library (GDS) – graph algorithms, ML support, graph administration
    - Centrality, community detection, Similarity, Path finding
    - Node embedding
  - GQL standard – ISO standard
    - built on openCypher, PGQL, GSQL, G-CORE

| RDF | x | LPG |
| --- | --- | --- |
| - global ids and IRI for | | - local labels and types |

RDF

- global ids and IRI for everything
- global vocabs
- focus on linking data across publishers
- statements need to become resources

LPG

- local labels and types
- all databases have independent types and labels
- best at graph algorithms

- RDF* - allows RDF statements to be objects / subjects of other RDF statements

Prezentace 6

pátek 20. ledna 2023 19:06

- XML - tags human-readable messages which allows queries
  - always a root element
  - document-centric XML - has root, similar to HTML document - meant for or by humans
- converting graph data into hierarchical - choose a node to "hang" and go down 1 level in hierarchy per relationship
- data-centric XML - autogenerated and regularized
  - created and consumed by apps
  - non-human readable without tags

XML declaration
root element
└ elements (can contain mix of texts and elements)
  ├ attributes
  └ nested elements

- case sensitive
- well-formed document - complies with XML syntax
- uses namespaces (using prefixes) - xmlns:prefix="prefix IRI"
  - don't need to be in the root, just root element of area in which it is used - qualified name
  - supports default namespaces xmlns="..."
- <![CDATA[everything here is a string]]>
- xml:lang tags
- processing instructions (PI) - not part of data
  - for XML parsers
  - such as stylesheet
- also allows entity/character references
- XML processing principles - Document object model (DOM) - loads entire XML to memory
  - poor for large files
  - poor for streams
  - supports querying
  - Simple API for XML (SAX) - processing as stream
    - reverse of DOM advantages
    - emits events on read
  - Streaming API (StAX) - same as SAX, but events are pulled by us, not emitted to us
- Parsing - DOM, SAX, StAX, LINQ
- Validation - Schema (XSD), RelaxNG ...
- Querying - Xpath
- Transformation - XSLT

- Persistence - databases
- Message transfer
- XML is rigorously standardized ekosystem, where JSON is a non-standardized object serializer
- RDF/XML is one of the first RDF serializations

specific formats (SVG, RSS, GO XML. ..) are also XML serializable

- XSD specifies structure of XML docs
- XSD supports typical basic data types
- eGovernments often uses encoded XML to Base 64
- XSD defines: data types - simple / complex
  elements - possibly groups
  attributes - possibly groups

- simple types don't have attributes or subelements
  - can have restrictions
  - can be a list
  - can be an union of simple types
  - can be enum
- complex types without subelements have simple content
  - text / attributes / no subelements
  - can have restrictions / extensions
- complex types with subelements have complex content
  - wraps subelements in sequence - order matters
    - or in choice / all
- schema can also use namespaces

Prezentace 7

sobota 21. ledna 2023 14:31

- XPath - queries resemble URLs with root element at the start and requested element/function on the requested element at the end
  - traverses XML as a tree of node types - document (root)
    - element
    - attribute
    - text
    - comment
    - namespace
    - processing instructions
  - traversal has no specific order
  - can access elements / their contents via functions / attributes
  - results can be filtered along the way by only targeting nodes with specific attributes / etc...
  - relative paths only need a starting point
  - axes - /root/descendant::title - gets all titles beneath root regardless of their depth
    - descendant / child / attribute / preceding-sibling / descendant-or-self / self / parent
    - in short, relationships to the current node, used to locate nodes relative to that node
    - functions like last(), position(), text(), count()...
  - logical expressions in [ ] after node name in path
  - supports sequences, cycles, conditional expressions, comments
  - mapping !
  - concatenation ||
  - function-chaining ⇒
- XSLT - transforms XML to HTML / XML / RDF Turtle / TXT
  - stylesheet - an XML document
    - set of templates
    - stylesheet root element
  - template - outputs text
    - matches parts of XML with XPath expressions
  - processor - matches templates to input XML
  - templates define - method - html / xml / ... } attributes of <xsl:output ...>
    - indent - yes / no
    - content
    - xsl: elements to process } <xsl: template ...>
    - select - XPath expression - <xsl: value-of ...>
  - apply-templates - applies a template rule to the current element or its children
  - implicit templates - present unless overriden
    - text from elements and attributes to be copied into output
  - named templates - name instead of match
    - accept parameters
    - xsl: call-template
  - global variables defined in stylesheet
  - mode - allows processing identical nodes in different modes

- global variables defined in stylesheet
- mode - allows processing identical nodes in different modes
  - specified in apply-templates
- supports switch, for each, include, import, streams, grouping, multiple outputs, regex, high-order functions, text processing (CSV, JSON..) on input

Prezentace 8

sobota 21. ledna 2023 16:47

- JSON - language-independent
  - easy to read and parse
  - often serialized
  - strings in UTF-8
  - only decimal numbers
  - values are separated by whitespaces
  - value can be - string
    - number
    - object - unordered set of name/value pairs, can be root
    - array - ordered collection of values, can be root element or a value
    - bool
    - null
  - several ECMA and RFC specifications
  - often used in web APIs
  - different data structures/standards
  - different cmd tools (jq)
  - JSON-based databases - Apache
  - JSON lines - each line is a valid JSON - alternative to CSV (.jsonl
  - XSLT: json-to-xml() or the other way around
  allows pointers to arbitrary values as strings or URI fragments (looks like XPath)
- JSON schema - annotation, validation, completeness
  - vocabulary annotating specific field and enforcing constraints
  - schema keywords/validation keywords: schema annotation
  - can assign properties to certain elements
  - validation can enforce types, requiredness, value ranges, number of occurrences, list contents
  - tuple validation - each list element can have a different schema
  - allows use of nested JSONs and external schemas
  - enables constraining value contents by length, regex, specific formats (for dates or IRI...)
  - validating against anyOf, allOf, oneOf, not schemas
  - schema extensions
  - can encode non-JSON data
- JSON-LD - makes JSON interpretable as RDF model by mapping via @keywords
  - for linked data
  - @context, @id, prop translates to IRI
  - @type specifies class IRI
    - can be an array
    - can be aliased
  - allows base and vocabulary specifications
  - values null in @context should not be in RDF
  - names defined in @context can shorten IRI like in RDF: foaf: some IRI, ..., foaf:name: something
  - contexts scoped by indentation

```
{
"@context":
        {
        @vocab: for @type
        @base: "IRI for blank ID"
        "name": "relationship IRI"
        },
"@id": "subject IRI",
"name": "object value",
"@type": ["name"]
"embedded": {
```

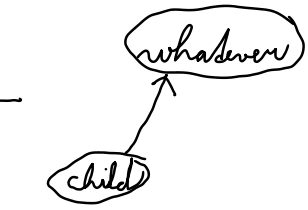
something

- contexts scoped by indentation
- @language
- if @context has @container: @bag,
  we can define a map named after id (in context)
  whose entries will all generate an extra
  node with corresponding value (one per
  language or whatnot)
- @list keyword preserves ordering

```
"foaf:listname": {
        "@list": [...]
        }
```

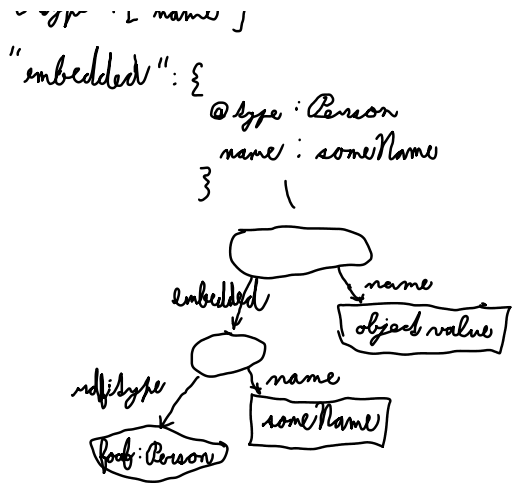
- @reverse allows us to change the relationship orientation

```
@context: {
    name: someVocab,
    children: {@reverse: vocab relationship}
    }
name: whatever,
children: [ {
    @id: ...
    name: ...
    }
...]
```




- @graph - named graph construction
- @id as val in context aliases key

@context can just contain a link to an external context



- relational data model - organized into rows and columns
  - id as primary key column
  - relationships via foreign keys
  - SQL dump - drops pre-existing data
    - creates new tables
    - loads data
- pre-CSV - delimiter-separated values (DSV) - delimiting - marks start and end
  - separator - separates values
  - can be combined
  - Tab-Separated Values (TSV) - records as rows
    - fields as columns
    - no tabs in values
    - headers present
    - constant number of tabs per line
- Comma-Separated Values (CSV) - uses CRLF
  - commas separate values
  - escape character
  - US-ASCII default
  - many editors either load or save CSV incorrectly
  - data types based on XSD XML schema
  - best practices - headers
    - null values are missing, rather than explicitly null
    - amounts and units should be in separate columns
    - no structured values in single cells
    - no print formatting
    - no sums of cells (they don't adhere to table schema)
  - one big CSV is better for ML and similar
  - several small files emulate a database better
  - can have HTTP content type header for web publication
- URL and CSV - columns and rows can be identified by corresponding URI fragments (ie. #col=2)
  * and cells
  - fragments can identify whole sections and multi-selections
  - errors - out-of-range selection - ignored
    - Range beyond table - trimmed
    - Inverse order - ignored
    - Ignored sections can still produce a valid fragment
  - in case client doesn't know RFC 7111, either ignores fragments or returns the whole file
- CSV on the web - CSV tables are annotated using JSON-LD descriptors
  - CSVW - annotated tabular data model
    - Table group, table, ... are main entities
    - name, title, schema, etc. are annotation properties
    - top-level properties are within @context - @base/@language
    - @base is used for URLs within metadata, URI templates don't use @base
    - prop "tables": [ ... ] annotates table group @type : TableGroup

- @base is used for URLs within metadata, URI templates don't use @base
- prop "tables": [ ... ] annotates table group @type: TableGroup
- @type: Table requires url property, can define tableSchema: { columns: ...
- supports all basic datatypes
- specifying schem further creates derived datatypes
- foreign keys are references to tables in the same group
- dialects - deviations from best practices
  - separators for lists of items within a cell
  - retaining of list ordering
  - explicit nulls
  - default values
  - formatted numbers/dates
    - implementations still have to support certain formats
- metadata location - user-provided / HTTP link header / canon URL endpoints with URI templates containing metadata/append -metadata.json to URL / csv-metadata.json relative URL / embedded metadata

- RDF from tabular data - RFC 6570 - URI Template: changing parts of URI
  - /~var/ - variable
  - /~{var}/ - expression
  - /~{+var}/ - var can contain percent-encoded triples
  - /~{#var}/ - fragment expansion
- JSON-LD - in tableSchema prop: - aboutUrl - translated to RDF subject
  - propertyUrl - predicate
  - valueUrl - object
- transformations (not in context)
  - defines how data is to be transformed
  - url - URL of script/template
  - scriptFormat - media type/URL
  - targetFormat
  - source - pre-transform
  - tables

Prezentace 10

neděle 22. ledna 2023 15:59

- Geodata - data depicting distance, elevation, size, even relative visibility
  - Geographic Information Systems = Geomatics
    - collection, distribution, storage, analysis, processing, and presentation of geodata
  - Geodata - associated with location relative to Earth
  - Coordinate ref. system - Reference ellipsoid + Datum + Projection
  - Implicit geodata - coordinates, distances, directions
  - Explicit geodata - references, address, geo. names
  - Geo. objects - Points, Multipoints, Lines, Multilines, Polygons, Multipolygons, Surface, w/o curves
  - representation - Well-Known Text (WKT) - Points (X whitespace Y), Linestring (X Y, X Y, X Y)
    - Geography Markup Language (GML) - similar to XML
      - described by a set of XSD
    - GeoJSON
    - Shapefile
    - GeoPackage
    - CSV
    - GeoSparql
- Spatial data logic - Feature → other attributes, ↘ Geometry
- Shapefile - Multiple table files
  - One feature per dataset
- GeoPackage - SQLite
  - geometry structures as attributes
  - vector and raster data
- CSV - can be formed into WKT

GeoWGS - another XML for pos. info and only points

- GeoSparql - Ontology + query language for spatial ops.
  - GML and WKT representation
  - supports CRS
- GeoJSON-LD
- Spatial relations - Topological / Directional / Distance / Temporal
- Spatial operations - Buffer / Union / Difference / Clip / Intersection / Interpolation / Distance / Convex Hull / Thiessen or Voronoi

- Graphical formats - raster/bitmap vs. vector
  - indistinguishable if rendered in the correct media resolution
  - bitmap resizing can blur the image (compression of pixel squares to smaller ones)
- Vector graphics - 2D - Points
  - Paths - Lines/Curves/Polygons
    - strokes/fill/color
  - Text - font/color
  - Scalable Vector Graphics (SVG) - web-browser friendly
    - XML-based
    - media type → image/svg+xml
    - embeddable to HTML
  - good for diagrams/slides/plots/graphs/plans
- Universal 3D (U3D) - Vertex-based
  - embeddable to PDF
- Raster graphics - Pixel/dot - colored square
  - pixel/dot matricies
  - Resolution - number of columns and rows
    - pixel amount
  - pixel density - dot/pixel per inch (DPI / PPI)
  - color - greyscale/monochrome/palletized/full color
  - RGB - red/green/blue numbers
    - additive for 3 "light sources"
    - mainly displays
  - CMYK - cyan/magenta/yellow/black
    - mainly printing
    - subtractive - how much of which ink
  - color space - all colors recognizable with eyes
    - polygon-shaped cutouts on the diagram represent specific spaces
    - standardizes hues of primaries
    - color model (like RGB) is mapped to color space
    - sRGB, AdobeRGB, DCI P3...
  - gamut - displayable colors
  - bit depth - number of colors (6 bits per color, 8, 16 ...)
  - transparency - RGBA - alpha channel
    - additional bits per pixel
  - dithering - achieving colors with available primary colors
    - more output pixels per input pixel
  - Temporal dithering - fast switching between neighboring colors of a pixel in time
  - BMP - Device-Independent Bitmap (DIB)
    - byte order by color (first Blue...)
  - Lossless compression - numbers od same-color adjecent pixels (possibly as 1 big array with connected rows)

- encoding by symmetrical blocks
- Quadtree encoding - splitting into quadrants of identical color
- Huffman coding - most common values into shortest bit sequences
- LZ77 - repeated vals replaced by references to first occurances
- GIFs, PNGs ...

- Lossy compression - discrete cosine transformation (DCT)
  - x is spatial, y is color
- quantization - discards high frequencies
- chroma subsampling - Y'CbCr - brightness, blue component, red c.
  color model
  - reduces resolutions of chroma components
- JPEG - raster-based
  - uses chroma sub.
  - splits image to blocks
  - DCT
  - quantization
  - lossless compression of results

- Raster formats (RAW formats) - typical contents - sensor metadata, thumbnail, metadata, raw sensor data
- transformations for image producing - decoding, demosaicing, defective pixel removal, white balancing, noise reduction, color translation, tone reproduction, compression

- Video formats - Uncompressed - 10 bpc RGB bitmap, each pixel is 32 bits
- Motion JPEG
- Compressed - inter-picture prediction - Marblocks - pixel blocks
  - associated motion vectors
- H.261 - 2 frame sizes
  - video conferences
  - DCT, quantization, inter-picture prediction, smoothing of marblock edges
- MPEG-1 - different frame types - I-frame - can be rendered independently
  - group of pictures
  - P-frame - predicted frames
  - B-frame - bipredictive frame - P-frame + difference from next frame
  - D-frame - low-quality DCT results
- H.262/MPEG-2, H.263 - interlaced video - odd and even pixel lines so double FPS
- more improvements increase numbers of versions
  - improve resolutions, compressions, motion vector precision

- video format-inspired raster formats - WebP - GIF/PNG/JPEG successor
  - open
  - lossless improvement on PNG
  - lossy JPEG improvement
- High Efficiency Image File Format (HEIF)

- digital audio - Pulse-code modulation (PCM) - samples analog signal

- quantized to nearest value from digital step range
- Linear pulse-code modulation (LPCM) - quantization is linearly uniform
- Waveform Audio File Format (WAV) - typically uncompressed
- Compact Disc Digital Audio
- lossless compression - FLAC - Linear prediction
  - Run length
  - Inter-channel correlation for Multi-channel audio
- lossy compression - MP3, AAC
  - discards sounds unhearable by humans (DCT)
  - Joint-stereo encoding - main channel - sum or avg L+R
    - side channel - L−R
  - Opus - open
    - highest quality

- Multimedia containers - Identification of different data types
  - Interleaving - of different data types or multiple streams (audio, video, subtitles...)
  - Metadata
  - Simple containers - JPEG, PNG, WAV, TIFF
  - Open or closed flexible containers
  - Matroska, WebM ...

- Print formats - PostScript (PS) - programming language
  - Encapsulated PS (EPS) - + bitmap preview
  - Portable Document Format (PDF) - PS-based
    - complete document desc. - metadata, contents, graphics
  - PDF/A - for archiving

Prezentace 12

neděle 22. ledna 2023 19:58

- SW configuration - .properties file - hash-table for java
  - localizations
  - also XML
- INI file - holds config
  - MS-DOS - replaced by Windows registry
    - W10 uses for desktop and php
  - human-readable and simple parsing
  - no nesting or standard
- Tom's Obvious, Minimal Language (TOML) - human readable and writeable
  - easy to parse
  - maps to hash table
  - native data types
  - Unicode
  - key-value and comments
  - strings and literals + multiline variants
  - other typical data types (even arrays and hash tables)
  - nesting
  - cloud HTTP proxy / YAML / Python deps.
- YAML - Unicode
  - Human and Machine-readable
  - lists, comments, maps, strings
  - types indicated by tags - URIs in 'tag:' scheme
  - schemas are sets of such tags and tag regex resolution rules

lifecycle: native → Node graph → Serialization (events) → Char stream