13 – BagIt

BagIt ist ein standardisiertes Containerformat auf Verzeichnisebene, welches beschreibt, wie man Dateien zur Speicherung und Übertragung digitaler Objekte verwenden sollte. Es trennt dabei strikt zwischen der eigentlichen intellektuellen Einheit (IE), im RFC ‚payload' genannt, und den beschreibenden Metadaten, die für die Übertragung minimal notwendig sind. Das BagIt Format ist sehr weit verbreitet und wird unter anderem als Archivformat in Archivematica verwendet.
Es wurde im Oktober 2018 als Version 1 im RFC8493 (https://tools.ietf.org/html/rfc8493) standardisiert.

Beispielhafter Aufbau

Wenn eine IE beispielsweise wie folgt aussieht:

```
exampleIE/
├── 1.txt
├── 3.dat
└── subdir/
    ├── 2.png
    └── 2.mdx
```

dann kann das resultierende Bag folgendermaßen aufgebaut sein:

```
examplebag/
├── bag-info.txt
├── bagit.txt
├── data/
│   ├── 1.txt
│   ├── 3.dat
│   └── subdir/
│       ├── 2.png
│       └── 2.mdx
├── manifest-md5.txt
├── manifest-sha512.txt
├── meta/
│   ├── mods.xml
│   └── rights.xml
├── tagmanifest-md5.txt
└── tagmanifest-sha512.txt
```

Ein Bag enthält mindestens die Bestandteile:

- bagit.txt – Eine Textdatei, die die Version und Zeichenkodierung des Bags beschreibt
- bag-info.txt – Eine Textdatei, die beschreibende Metadaten in Form von Schlüssel-Wertepaaren enthält, einige sind zwingend erforderlich, andere optional. Zu den zwingend erforderlichen gehört zum Beispiel Payload-Oxum, welches die Gesamtzahl an Bytes und Gesamtzahl an Dateien im Payload-Verzeichnis angibt.
- data/-Verzeichnis, welches die Dateien des IE enthält (Payload)
- manifest-sha512.txt Datei, welche die Prüfsummen der Payload-Dateien enthält

Optional sind in diesem Fall die tagmanifest-Dateien. Wie man in dem Beispiel sieht, kann man eigene Verzeichnisse definieren, deren Dateien nicht zur Payload gehören, die man aber über die tagmanifest-Dateien referenzieren kann. Diese werden von BagIt-Programmen in der Regel nicht ausgewertet.
Die BagIt-Spezifikation gibt vor, welche weiteren Dateien für Bags reserviert sind, und beschreibt, wie die Manifest- und Tagmanifest-Dateien aufgebaut sind.
Grundsätzlich ist es möglich unicode-kodierte Dateinamen zu verwenden, man sollte in dem Fall aber ebenfalls auf die Hinweise im RFC achten.

Nachfolgend sind einige der im Beispiel verwendeten Dateien aufgelistet:

Inhalt der *‚bagit.txt'*:
BagIt-Version: 1.00
Tag-File-Character-Encoding: UTF-8

Inhalt der *‚baginfo.txt'*:
Author: Max Mustermann
Bagging-Date: 2015-12-28
Bag-Size: 389 kB
External-Description: Dies ist ein kleines Beispiel für
eine IE, die als SIP im BagIt-Format eingeliefert
werden soll.
External-Identifier: testbag-01
Payload-Oxum: 388743.4
Title: BeispielIE

Inhalt der *‚manifest-md5.txt'*:
d41d8cd98f00b204e9800998ecf8427e data/subdir/2.png
227bc609651f929e367c3b2b79e09d5b data/subdir/2.mdx
e1cbb0c3879af8347246f12c559a86b5 data/1.txt
d41d8cd98f00b204e9800998ecf8427e data/3.dat

Inhalt der *‚tagmanifest-md5.txt'*:
d879078aefbe30394bf4fbf602daaae2 manifest-sha512.txt
7cb81a65369e7bbb9794971f2f1baeff manifest-md5.txt
9ca92ea8aabf7c50b61d24422748f1ad bag-info.txt
eaa2c609ff6371712f623f5531945b44 bagit.txt
63d20275d78e2dfddc135d8c6acbae15 meta/rights.xml
d41d8cd98f00b204e9800998ecf8427e meta/mods.xml

Vorteile von BagIt

Im Vergleich zu METS-basierten Containern bietet BagIt eine Reihe an Vorteilen. Zwei Vorteile springen dabei ins Auge. Erstens sind Bags menschenlesbar. Dies erleichtert die Erfassung und die Fehlersuche.
Zweitens erlaubt die komponentenorientierte Aufteilung in verschiedene Dateien und insbesondere die strikte Trennung zwischen "Payload" und eigentlichen Metadaten die einfachere Weiterverarbeitung. Oftmals ist es daher möglich, Bags mit Standardwerkzeugen zu verarbeiten.
BagIt definiert zudem die Verwendung von integritätssichernden Prüfsummen, wie SHA512.

Verwendung

BagIt erfährt in den letzten Jahren zunehmend Verbreitung als Grundlage von Archivinformationspaketen, zum Beispiel in dem AIS-System Archivematica.
Auch im Ingest als Einlieferungspaket wird es verwendet. So hat die SLUB Dresden ihre Spezifikation mit dem Release 2020.1 von METS-basierten SIPs auf BagIt umgestellt. Neben Archivematica versteht auch die Archivsoftware Rosetta BagIt basierte SIPs .

Weiterführende Informationen

Die Unterstützung von BagIt ist unterschiedlich gut ausgeprägt. Noch sehr verbreitet ist die Version 0.97. Mit Archive::BagIt (https://metacpan.org/pod/Archive::BagIt) steht eine Perl-API zur Verfügung, die auch die Version 1.0 nach RFC 8493 gut unterstützt. APIs stehen auch für andere Programmiersprachen (Python, Java, C++) zur Verfügung. Mit Bagger (https://github.com/LibraryOfCongress/bagger) existiert eine java-basierte GUI-Software der Library of Congress, die leider nicht mehr aktiv gepflegt wird. Eine gute Übersicht über APIs und Programme liefert der Wikipedia-Artikel zu BagIt unter https://de.wikipedia.org/wiki/BagIt#Implementierungen.

Der RFC 8493 ist unter https://tools.ietf.org/html/rfc8493 frei verfügbar.

Unterschiede zwischen 1.0 und 0.97

- In 1.0 wird standardmäßig SHA512 als Prüfsummenalgorithmus empfohlen, 0.97 verwendet dagegen MD5.
- In 1.0 müssen konforme Programme Testsuiten implementieren.
- In 1.0 sind Dateinamen und deren Mapping genauer spezifiziert.
- In 1.0 ist die Serialisierung von Bags in Kompressions- und Archivierungsdateiformate wie ZIP oder TAR nicht mehr Bestandteil der Spezifikation.

Glossar

- AIS – Archivinformationssystem, Software zur Verwaltung eines digitalen Archivs
- API – Programmierschnittstelle bzw. Programmbibliothek
- Bag – Verzeichnisstruktur nach BagIt-Spezifikation
- IE – intellektuelle Einheit, zusammengehörende Dateien, die ein digitales Objekt formen
- Manifest-Datei – Textdatei, die Prüfsummen und Dateinamen der Payload enthält
- METS – Metadata Encoding & Transmission Standard, XML-Dateiformat
- Payload – Nutzlast, in diesem Fall gleichbedeutend der IE
- RFC – request for comments, Bezeichnung für eine Spezifikation der IETF Standardisierungsorganisation
- Tagmanifest-Datei – Textdatei, die Prüfsummen und Dateinamen aller Bag-Dateien mit Ausnahme von Tagmanifest-Dateien und Payload-Dateien enthält

Andreas Romeyke

Mitarbeiter an der Sächsischen Staatsbibliothek, Landes-und Universitätsbibliothek Dresden (SLUB).
Kontakt: mailto:romeyke@slub-dresden.de, Webseite: http://andreas-romeyke.de

Romeyke beschäftigt sich mit der Analyse von Dateiformaten und Workflows im Rahmen der digitalen Langzeitarchivierung. Mit seinem Kollegen Jörg Sachse schreibt er regelmäßig zum Thema digitale Langzeitarchivierung auf dem gemeinsamen Blog "Kulturreste" unter https://kulturreste.blogspot.de.

Weitere Kurzartikel aus der Reihe „nestor Thema" finden Sie auf www.langzeitarchivierung.de -
der Webseite von nestor – Kompetenznetzwerk Langzeitarchivierung.