

- [Hochladen von Dateien in eine Sharepoint-Bibliothek-in-Canvas-Apps](#)
- [Referenz](#)
- [Teil 1 - eine Datei hochladen](#)
- [Teil 2 - mehrere Dateien hochladen](#)
- [Teil 3 - Metadaten bearbeiten](#)
 - [Hier die Code Ansicht vom Flow:](#)
 - [Hier der Aufruf in PowerApps:](#)

Hochladen von Dateien in eine Sharepoint-Bibliothek-in-Canvas-Apps

Dieses Videos beschreiben wie man Dateien in einer Sharepoint Bibliothek von einer Canvas App innerhalb der Standard Lizenz hochladen kann.

Referenz

basierend auf dem Blog Post von Matthew Devaney (Prädikat besonders empfehlenswert)

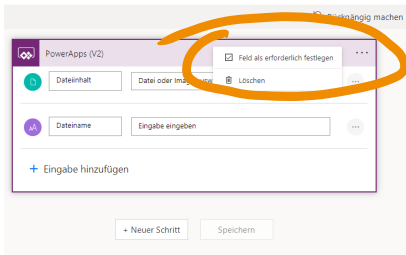
<https://www.matthewdevaney.com/power-apps-easiest-way-to-upload-files-to-a-sharepoint-document-library/>

Teil 1 - eine Datei hochladen

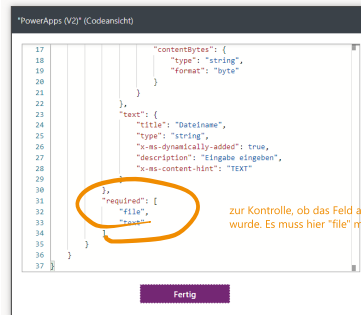
Achtung: der Flow muss im Classic Designer angelegt werden, da ansonsten der File Parameter als optional definiert wird und der Aufruf von PowerApps aus nicht funktioniert. Im neuen Designer kann diese Option derzeit noch nicht gesetzt werden.

Link zum Video: https://youtu.be/AbRa6_sgqTg

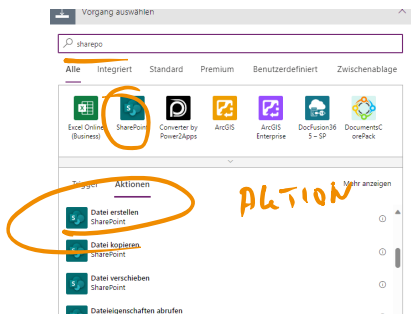
1. Sharepoint vorbereiten
Bibliothek und eine Liste
2. Power Automate Flow zum Hochladen erstellen



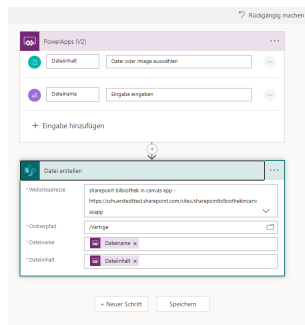
es muss diese Option "Feld als erforderlich festlegen" sichtbar sein
sollte hier noch "Feld als optional festlegen" stehen, dann durch Klick umstellen.



zur Kontrolle, ob das Feld als erforderlich markiert wurde. Es muss hier "file" mitaufgeführt werden

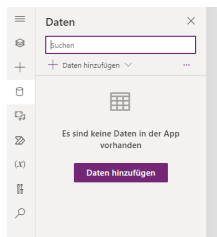


Aktion



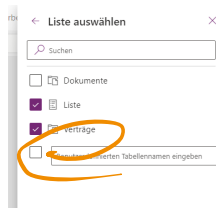
fer fertige Flow

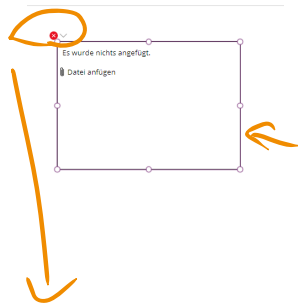
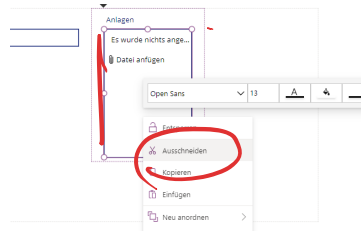
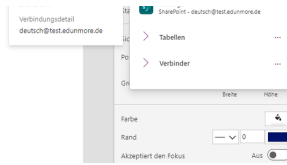
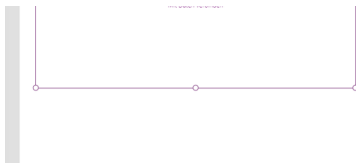
3. Power Apps Canvas App konfigurieren



Sharepoint als Datenquelle konfigurieren - wir brauchen die Bibliothek und eine beliebige Liste!

L

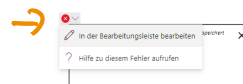




Das ausgeschnittene Anlagen-Element zeigt Fehler an.

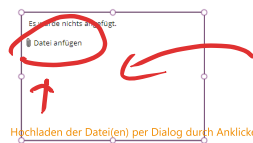
Wir müssen alle Fehler beheben, damit das Element dann funktioniert.

Am Einfachsten immer wieder auf das rote Fehlerelement klicken und nach und nach alle Fehler beheben.



Hier ein paar mögliche Fehler - mehr im Video:

```
Items: Blank()
DisplayMode: DisplayMode.Edit
MaxAttachments: 1
```

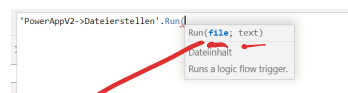


Das Element unterstützt das Hochladen per Maus - Drag&Drop

Hochladen der Datei(en) per Dialog durch Anklicken



Die Dateien werden temporär gespeichert. Um die Dateien in die Sharepoint Bibliothek zu laden, müssen wir den Power Automate Flow aufrufen. Hier im Beispiel über eine Schaltfläche.



```
UploadFileToDocumentLibrary.Run(
{
  contentBytes: First(attach_SubmitContract_AttachFile.Attachments).Value,
  name: First(attach_SubmitContract_AttachFile.Attachments).Name
});
```

Der Dateiparameter ist ein Objekt mit den Parametern contentBytes und name.

```
'PowerAppV2->Dateierstellen'.Run(
{
  contentBytes: First(Anlagen.Attachments).Value;
  name: First(Anlagen.Attachments).Name
});
```

First holt den ersten Datensatz aus der Tabelle

Attachments Tabelle



Zeilen der Tabelle
Value : der Dateiinhalt
Name: der Dateiname

Anlagen (Name des Elementes in PowerApps)

Tip: wenn der File Parameter nicht als Erforderlich festgelegt wurde, kann man diesen als optionalen Parameter übergeben im Format:

als optionalen Parameter übergeben im Format:

```
{File:
  {contentType: ...
    name: ...
  }
}
```

Teil 2 - mehrere Dateien hochladen

Link zum Youtube Video <https://youtu.be/5WV59Xq5v3c>

Schleife mit ForAll über die Tabelle als anhang Variable in der Schleife

im Beispiel sind drei Datensätze
daher werden 3 Flows in Power Automate gestartet
pro Datensatz/Datei ein Flow

anhang.Value und anhang.Name enthalten die Daten pro Durchlauf

Zusammenfassung

dynamischer Text im Anlagen Element der nach dem Hochladen eine Meldung anzeigt

```
UpdateContext({txtAnlagenLog: "Keine Dateien angefügt!"})
```

GALLERY

Launch(ThisItem, "Link zu Element")

transparentes AnlagenElement mit einer Icon Boxkammer im Hintergrund zur besseren Option

Hochladen per OnAddFile Event (siehe Kommentare im Code)

```
OnAddFile = fx: ForAll(
  Anlagen_2.Attachments As anhang; // Schleife über alle Anlagen
  'PowerAppV2->Dateierstellen'.Run( // pro Durchlauf wird ein Anhang (Attachments) in der Variable anhang gespeichert
  { // Aufruf des Flows
    contentBytes: anhang.Value; // File Parameter braucht contentBytes und name
    name: "" // der Dateiname - aus den Attachments
  }; // den ignorieren wir hier und geben den Namen als gesonderten Parameter an
  anhang.Name // der Dateiname aus den Attachments
);
UpdateContext({txtAnlagenLog: "Datei(en) wurden gespeichert!"}); // Text nach Anzeige eines Hochladens
Reset(Anlagen_2); // alle Anhänge auf dem Kontrollelement entfernen
Refresh(Verträge); // Gallery neu anzeigen
dev_o true // Rückgabewert des Events - wird m.E. aber nicht ausgewertet
dev_o
```

Skript zum Hochladen:

```
ForAll(
  Anlagen_2.Attachments As anhang; // Schleife über alle Anlagen
  'PowerAppV2->Dateierstellen'.Run( // pro Durchlauf wird ein Anhang (Attachments) in der Variable anhang gespeichert
  { // Aufruf des Flows
    contentBytes: anhang.Value; // File Parameter braucht contentBytes und name
    name: "" // der Dateiname - aus den Attachments
  }; // den ignorieren wir hier und geben den Namen als gesonderten Parameter an
  anhang.Name // der Dateiname aus den Attachments
);
UpdateContext({txtAnlagenLog: "Datei(en) wurden gespeichert!"}); // Text nach Anzeige eines Hochladens
Reset(Anlagen_2); // alle Anhänge auf dem Kontrollelement entfernen
Refresh(Verträge); // Gallery neu anzeigen
true // Rückgabewert des Events - wird m.E. aber nicht ausgewertet
```

Teil 3 - Metadaten bearbeiten

Link zum Youtube Video <https://youtu.be/kgTa8DOIUuI>

Im 3. Teil schauen wir uns an, wie wir den Flow dahingehend ändern, dass Metadaten gespeichert werden.

Änderungen am Flow:

ACHTUNG: optionale Parameter werden derzeit nur im Classic Designer angezeigt!

als optional festlegen
(es muss die Option so angezeigt werden)

hier werden die erforderlichen Parameter in der Code-Ansicht angezeigt
Als Bezeichner werden aber die internen Namen verwendet - hier: file und text für die ersten beiden Parameter im Flow.

Aufruf des Flows von PowerApps:

```
'PowerAppV2->Dateierstellen'.Run(  
  {  
    contentBytes: First(AnLagen.Attachments).Value;  
    name: ""  
  });  
First(AnLagen.Attachments).Name;  
  {  
    text_1: "eine Demo";  
    date: "2024-01-16"  
  }  
);;
```

optionale Parameter werden als Objekt () übergeben
Namen der Parameter kann man im Flow Code sehen - es werden hier leider nicht die von gewählten Namen verwendet.

Hier die Code Ansicht vom Flow:

```

{
  "kind": "PowerAppV2",
  "inputs": {
    "schema": {
      "type": "object",
      "properties": {
        "file": {
          "title": "Dateiinhalt",
          "type": "object",
          "x-ms-dynamically-added": true,
          "description": "Datei oder Image auswählen",
          "x-ms-content-hint": "FILE",
          "properties": {
            "name": {
              "type": "string"
            },
            "contentBytes": {
              "type": "string",
              "format": "byte"
            }
          }
        },
        "text": {
          "title": "Dateiname",
          "type": "string",
          "x-ms-dynamically-added": true,
          "description": "Eingabe eingeben",
          "x-ms-content-hint": "TEXT"
        },
        "text_1": {
          "description": "Eingabe eingeben",
          "title": "Titel",
          "type": "string",
          "x-ms-content-hint": "TEXT",
          "x-ms-dynamically-added": true
        },
        "date": {
          "description": "Geben Sie ein Datum ein, oder wählen Sie ein  
Datum aus (JJJJ-MM-TT).",
          "format": "date",
          "title": "Verkaufsdatum",
          "type": "string",
          "x-ms-content-hint": "DATE",
          "x-ms-dynamically-added": true
        }
      }
    },
    "required": [
      "file",
      "text"
    ]
  }
},
{
  "metadata": {
    "operationMetadataId": "30aa4fd9-7a3f-47bc-bf10-4d7b88045e8a"
  }
}

```

```
}  
}
```

Hier der Aufruf in PowerApps:

```
'PowerAppV2->Dateierstellen'.Run(  
  {  
    contentBytes: First(Anlagen.Attachments).Value;  
    name: ""  
  };  
  First(Anlagen.Attachments).Name;  
  {  
    text_1: "eine Demo";  
    date: "2024-01-16"  
  }  
);;
```