

Forms That Survive Save and Reopen

A seven-field equipment-loan case from naming to reader acceptance

01

02

03

How to use this guide

Coordinators creating small reusable AcroForms and checking saved values.

Design a simple fictional loan form, expose duplicate identities and overflow, and record actual save-and-reopen behavior without assuming universal reader support.

- Specify independent fields, labels and valid states.
- Detect the supplied duplicate and sequence defects.
- Run a controlled save, update and reopen trial.
- Distinguish editable forms from checked viewing copies.

Scope and limitations

- The optional PDF generator requires Python 3 and ReportLab; the standard-library specification checker does not.
- Generated practice fields use basic Latin text and are not an accessibility-certified template.
- No electronic-signature, legal agreement, automatic submission or universal reader compatibility is promised.

Published by KitForma. Edition 1.0. Sources checked 1 October 2026. People and businesses in worked examples are fictional. This is an independent publication, not endorsed by the platforms mentioned.

Copyright 2026 KitForma. Usage, support and refund terms are included in LICENSE.txt in the purchased package.

Contents

How to use this guide	2
Contents	3
Define the equipment-loan task	4
Sources and tools	6

Define the equipment-loan task

A volunteer at the fictional Beacon Tool Room needs to record who collected a projector, which accessories went with it, and when it should return. The form travels as a file: the borrower opens it, enters values, saves a copy, closes the reader, and sends the saved copy back. That journey is the product. A beautiful blank page that loses its values after closing is an unsuccessful form, even if every border aligns. This workbook follows that journey through a small, inspectable AcroForm exercise.

Keep the scope modest. The practice form is an administrative record, not a rental contract, identity check, digital signature system, or proof that a person accepted legal terms. It contains fictional values and no submission button. A typed name records text; it does not demonstrate who typed it. If your real process requires identity verification or enforceable agreements, arrange the appropriate process separately. Adding a checkbox labelled agree does not establish those properties, and adding more fields does not make the document more trustworthy.

Write down seven data items before drawing boxes: borrower name, contact note, equipment reference, return date, carrying-case checkbox, cable checkbox, and notes. Each must serve the checkout task. The contact note is a fictional internal routing instruction rather than a real email address or phone number. This keeps the exercises harmless while still testing separate values. The date has an explicit year-month-day example so readers do not have to infer the order of ambiguous numbers. Notes are optional and deliberately bounded.

The supplied JSON specifications, field ledger, test cases, and generator are the practice materials. The generator requires Python 3 and the free ReportLab package; it creates a simple PDF with standard fields for your

reader tests. It is not a tagged, accessibility-certified production form and does not install software automatically. You may instead build the specified fields in an editor you already have. Either route must face the same save-and-reopen checks. The book's main deliverable is a defensible form test, not a claim that every application behaves identically.

Sources and tools

[S1] Adobe: field properties for form components

<https://helpx.adobe.com/acrobat/desktop/work-with-pdf-forms/customize-form-fields/field-properties.html>

Checked: 2026-10-01

[S2] W3C WAI forms tutorial

<https://www.w3.org/WAI/tutorials/forms/>

Checked: 2026-10-01

[S3] W3C PDF3: tab and reading order

<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF3>

Checked: 2026-10-01

Inside the complete guide

Separate names, labels and values

Choose controls that match the question

Design the page around reading and entry

Use keyboard order as a practical review

Make long text a deliberate failure case

Save, close and reopen the actual copy

Decide when an editable copy becomes a viewing copy

Write a compatibility report that helps the next person

Capstone and answer key