```
# check_xibs_font_outlets.py
from pathlib import Path
import xml.etree.ElementTree as ET

# ------------------------
# Replace this with your own list_xibs() output if you have it:
# e.g.
# from your_list_module import list_xibs
# xib_files = list_xibs()
# ------------------------
# Minimal example / placeholder:
xib_files = [
    # "/absolute/path/to/Project/App/UI/MyView.xib",
    # "/absolute/path/to/Project/Views/SomeCell.xib",
]

# If you already have a list_xibs() in the same script, comment the placeholder above and
uncomment below:
# xib_files = list_xibs()

# ------------------------
# Helpers
# ------------------------
def _strip_tag(t: str) -> str:
    """Strip namespace from an element tag."""
    return t.rsplit('}', 1)[-1] if '}' in t else t

def _collect_outlet_destinations(root: ET.Element) -> set:
    """
    Find all destination ids referenced by <outlet ... destination="..."/>
    and <outletCollection ... destination="..."/> across the XIB.
    """
    dests = set()
    for elem in root.iter():
        tag = _strip_tag(elem.tag)
        if tag in ("outlet", "outletCollection"):
            dest = elem.attrib.get("destination") or elem.attrib.get("destinationId")
            # older/varied xib might use 'destination' attribute; fall back to 'property'?
            if dest:
                dests.add(dest)
    return dests
```

```
def _element_uses_font(elem: ET.Element) -> bool:
    """
    Heuristic: an element uses a font if anywhere inside it there's a <fontDescription> or <font>
tag.
    This covers UILabel, UIButton titleLabel, UITextField, UITextView, and other IB font encodings.
    """
    for child in elem.iter():
        tag = _strip_tag(child.tag)
        if tag in ("fontDescription", "font"):
            return True
    # also check some attributes (defensive)
    for k in elem.attrib:
        if "font" in k.lower():
            return True
    return False

def _find_font_elements_without_outlet(root: ET.Element) -> list:
    """
    Return list of tuples (elem_tag, elem_id, elem_userLabel) for elements that use fonts
    but whose id is not referenced by any outlet in the file.
    """
    missing = []
    outlet_dests = _collect_outlet_destinations(root)

    # Search likely view elements (but we check all - if they use a font -> test them)
    for elem in root.iter():
        if _element_uses_font(elem):
            elem_id = elem.attrib.get("id")
            # If element has no id at all, treat it as "no outlet"
            if not elem_id:
                missing.append((_strip_tag(elem.tag), None, elem.attrib.get("userLabel")))
                continue
            # If id exists but is not referenced by any outlet -> missing outlet
            if elem_id not in outlet_dests:
                missing.append((_strip_tag(elem.tag), elem_id, elem.attrib.get("userLabel")))
    return missing

# ------------------------
# Main check routine
# ------------------------
```

```
def check_xibs_for_fonted_views_without_outlets(xib_paths):
    """
    For each xib path, print its path if it contains a view using a font that has no outlet.
    Returns a dict {xib_path: list_of_missing_elements} for further programmatic use.
    """
    report = {}
    for p in xib_paths:
        path = Path(p)
        if not path.exists():
            print(f"Skipping (missing file): {p}")
            continue
        try:
            tree = ET.parse(str(path))
            root = tree.getroot()
        except ET.ParseError as e:
            print(f"XML parse error in {p}: {e}")
            continue

        missing = _find_font_elements_without_outlet(root)
        if missing:
            # print xib name (user asked for filename)
            print(path.as_posix())
            # optionally print details (comment/uncomment as needed)
            for tag, eid, ulabel in missing:
                print(f"  - element: <{tag}> id={eid!s} userLabel={ulabel!s}")
            report[str(path)] = missing
    return report

# -----------------------
# Run when executed directly
# -----------------------
if __name__ == "__main__":
    if not xib_files:
        print("No xib_files defined. Assign xib_files = list_xibs() or hardcode paths at top of the
script.")
    else:
        check_xibs_for_fonted_views_without_outlets(xib_files)
```