Run tests and access private networks

Run tests and verify the first session

Interactive WebDesigner and saved scripts

1. Apply the configuration to the workstation that runs DSServer, then refresh its browser list.
2. Choose the enabled BrowserStack-… profile in the normal browser selector.
3. Incognito does not matter. Every BrowserStack session already starts with a fresh, isolated profile, so DSServer ignores the incognito setting for BrowserStack profiles (since 17 Nov 2026; older builds refused it with BROWSERSTACK_UNSUPPORTED: explicit incognito mode). Selecting a browser in WebDesigner only selects it; it changes no settings.
4. Begin with a short script: navigate to the application, wait for a known element, perform a representative click or form entry, and assert the expected page state.
5. Stop the interactive session when finished. Saved-script lifetime follows the existing script owner; verify cleanup rather than assuming a completed action closes every session.
6. Find the matching build/session in Automate using the complete AIQ-… run name. Confirm the target and ended state, and compare application assertions with the DS playback result.
7. If the browser does not start, the run fails with the reason. WebDesigner's execution log shows Could not start <browser>: <reason>, for example BROWSERSTACK_REFUSED: Automate testing time expired., and the page returns to idle instead of staying in "loading". Saved scripts and AIQ scenarios get the same message as the DSClient start error.

AIQ scenarios

In the scenario editor select the profile as the testcase **Browser** or **Override Browser**. AIQ delivers the current private provider configuration separately to the execution node. Credentials are not serialized into the scenario. The launch bridge maps its default incognito preference to normal cloud startup.

Install matching components on every node, confirm application/VPN reachability from those nodes, and run a short scenario on each intended node before scheduling a larger workload. **Distributed BrowserStack scenario acceptance remains pending in the reviewed evidence.**

First-run acceptance checklist

- The intended profile is visible and its account check succeeds.
- The reported browser, OS and device match the intended target.
- Navigation, element lookup, interaction and an application assertion complete.
- The BrowserStack session is ended and the DS log confirms remote stop.
- No unexpected cloud session or owned Local process remains after the run.
- If enabled, accessibility reports saved=true and a final report is available; a functional pass alone does not establish a completed accessibility scan.

The default provider idle timeout is 90 seconds. Long pauses can end the browser. A session-start request is not automatically retried: after an uncertain allocation DSServer attempts to identify and clean up only the exact owned session. Diagnose the original run before starting another.

BrowserStack Local and private network access

DSServer starts and stops a private BrowserStack Local process for each cloud run, using the run's unique identifier and assigned DS proxy. Keep this managed path for Appvance execution. The generic BrowserStack instructions for manually running a Local binary are useful for understanding networking, but do not replace the session-owned DS tunnel.

Requirement	How to prepare
Application access	From the actual DSServer machine, verify the application URL, DNS, VPN and required authentication routes. A VPN available only on a developer laptop does not give a separate execution node access.
Outbound provider access	Allow HTTPS/WSS to BrowserStack, normally outbound 443. BrowserStack's firewall/VPN guide specifies *.browserstack.com on 443. Coordinate allowlisting with the network team.

Requirement	How to prepare
Session and account services	Reach hub-cloud.browserstack.com and api.browserstack.com.
Local updates	Reach local-downloads.browserstack.com; verify the DSServer account can write its managed cache/runtime locations.
Accessibility services	Reach collector-observability.browserstack.com and api-accessibility.browserstack.com, or the configured regional report API.
Complete installation	Keep the release-matched browserstack-local folder beside the DSServer JAR. Relative paths resolve from that JAR folder.

The bundled host choices are Windows x64, macOS x64/ARM64, Linux x64/x86/ARM64 and Linux musl x64. DSServer chooses the binary for the **execution machine**, independently of the cloud browser's OS. Windows execution is recorded; foreign binaries were downloaded and format/hash checked, not executed in the cited provisioning checks.

Managed startup checks official downloads and can reuse a verified cached/bundled copy if an update fails. Each run uses its own writable runtime copy. The normal cache is ~/.appvance/browserstack-local. See BrowserStack's Local downloads for provider distributions.

Safari HTTPS: keep the supplied Safari/iOS networkLogs=true policy and captureNetworkContent=false. The provider's network-log proxy is part of the measured certificate-trust solution. Response-body capture is disabled, but network metadata may still be recorded. acceptInsecureCerts alone did not establish trust in the recorded iOS checks.

Revision #1
Created 2026-11-17 17:44:03 UTC by Admin
Updated 2026-11-17 17:44:03 UTC by Admin