New Relic Mobile Best Practices
Tips and Tricks for Improving App Performance and Stability

Fast, stable apps make for happy, productive users. The right data and tools to debug crashes and slowness across the stack make happy mobile and operations engineers. Here are some best practices you can employ to take full advantage of New Relic Mobile to eliminate crashes and increase speed across the stack—in the process building better performance into every app release.

1. **Start collecting data in production**
   
   You’ll see immediate value by adding the New Relic Mobile agent to your app for its next release to the app store. We recommend adding the agent to your production release even during your initial trial of New Relic as it will ensure you’ll get an adequate amount of data to really understand all the features of New Relic Mobile and your end users’ experience.

   **HOW TO DO IT**

   1. Add a new app in New Relic Mobile’s user interface.
      - Go to rpm.newrelic.com/mobile.
      - From the Mobile Apps index, select **Add a new app**.
      - Type a name for your mobile app, and select **Continue**.
      
      **Note:** Use one app and the associated app token for the development releases of your iOS app, another app and its associated app token for the production releases of that iOS app, and another pair of app tokens for development vs. production releases of your Android app. You will be able to compare data from across the four app tokens using a New Relic Insights dashboard (available for free as part of New Relic Mobile Enterprise).

   2. Download the latest version of the New Relic SDK and add it to your mobile app.
      - To take advantage of MobileRequest and MobileRequestError event data for error rates and response times of specific endpoints, you will need iOS and/or Android agent releases 5.14 or later. (Full steps iOS, full steps for Android)

   3. Configure your installation as needed. For example:
      - Enable the feature flag to capture MobileRequest events to access the full network analysis feature. (iOS/Android)
      - Automate dSYM uploads for your production builds.
      - To avoid surprises, follow the standard pattern of where you start the New Relic agent. In addition, you need to start the New Relic agent before you can initiate any other SDKs. (Android Install and Configure, iOS Install and Configure) If you are pondering a different approach, contact https://support.newrelic.com.

   4. Deploy a new release of your application.

   5. More information, consult the Mobile APM documentation or go to the Install and Configure page.

2. **Explore key features**

   Take the time to explore the following features as soon as the agent starts reporting data.

   - **Crash analysis.** Go to our Crash Analysis UI page to analyze crash data so that you can see and fix the most impactful crashes.
   - **Crash event trails.** Use the Crash Event Trail tab in the Crash Analysis UI to fix crashes more easily by using the breadcrumbs leading up to specific crashes.
   - **HTTP request performance.** Borrow from our example Insights queries to filter to end points for HTTP error rates and response times. Filter out uninteresting HTTP error status codes. See response time by attributes (such
Focus on mobile crashes and network performance

Once you are running New Relic Mobile in production and have explored some of its key features, you can use it to uncover critical issues with your mobile application—focusing on crashes and network performance.

Use metric-based alerts as a starting point to gain high-level visibility into problems. Then, as new services or functionality are added, use New Relic’s more targeted, event-based (NRQL) alerts to make sure that the most critical network requests are handled swiftly and have a low error rate (per end point). You can also use these alerts to determine whether a crash is impacting a large number of users. If it is, the relevant people—likely both the mobile app team’s on-call member as well as the owner of the related back-end service—will be alerted.

Mobile crashes

Make sure that your team knows how to respond to changes in crash rate (crashes/sessions) and the percentage of users impacted by crashes of your mobile app.

HOW TO DO IT

Level 1. Leverage our metric-based alerts and New Relic Mobile UI.

1. Ensure that key members of your mobile team have opted in to receive “new crash” emails and “version trends” emails. These emails will take you to the crash type in either the Mobile UI page or Version Trends UI page. (“new crash” emails, Version Trends emails)

2. Create a crash-rate-threshold alert that key members of your team can view in Slack (or another Alerts notification channel of your choice). This alert will take you to the Alerts UI to view a chart with the exceeded threshold, and a link will take you from there to the relevant UI page in New Relic Mobile. (more)

Level 2. Combine the metric-based workflow with event-based alerts and Insights dashboards specific to your mobile app and system.

1. Create an NRQL alert that focuses on crashes of your most recently released production version.

   • Example query.
   
   ```sql
   SELECT percentage(uniqueCount(sessionId),
   WHERE sessionCrashed is true) FROM Mobile-
   Session WHERE appName = 'Your App Name'
   AND appVersion = 'AppVersion'
   ```

Figure 1. Crash analysis screen.
2. Create an NRQL alert that focuses on the percentage of users impacted by crashes in your latest app version (to check the overall impact on your user base).

   • Example query.

```
SELECT percentage(uniqueCount(uuid), WHERE sessionCrashed is true) FROM MobileSession WHERE appName = 'Your App Name' AND appVersion = 'AppVersion'
```

4. NRQL alerts: Define thresholds for HTTP response times of your most critical end points.

5. NRQL alerts: Focus on error rates for specific end points using a combination of MobileRequestError and MobileRequest events to calculate your error rate filtered to your business-critical endpoints, using "wild-carded" URLs.

6. Set up dashboards to display necessary alerts and good thresholds, and to diagnose problems and enable cross-team communication in the case of an alert.

7. Ensure that you’re running Version 5.14 or later of the Android and iOS agents.

8. Use data from HTTP request response time for third-party conversations.

9. Example MobileRequest and MobileRequestError event queries

Download New Relic’s mobile apps for alerts and access to New Relic dashboards from wherever you are. (More)

---

**Figure 2. Version Trends email.**

**New Relic Demo - Production 1.2.3 (Android)**

Comparing launch week performance to previous version

<table>
<thead>
<tr>
<th></th>
<th>1860</th>
<th>186</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>App launches</strong></td>
<td>+19%</td>
<td>+36%</td>
</tr>
<tr>
<td><strong>Total Users</strong></td>
<td>+50 new instals</td>
<td>+130 upgrades</td>
</tr>
</tbody>
</table>

Crash rate: 0%
HTTP error: 0.33%
Network failures: 0.4%

Figure 3. HTTP Error dashboard.

**Mobile network performance**

Help your team understand and respond to changes in HTTP error rate, percentage of users impacted by HTTP errors, network failure rate, percentage of users impacted by network failures, and response time.

**HOW TO DO IT**

**Level 1.** Leverage New Relic metric-based alerts and New Relic Mobile UI. An alert will take you to the Alerts UI to see a chart with the exceeded threshold, and a link will take you from there to the relevant UI page in New Relic Mobile.

1. Create a general HTTP error rate and network failure rate alert. (More)

2. Create HTTP error and network failure rate alerts for specific host names. (More)

**Level 2.** Combine the metric-based workflow with event-based alerts and Insights dashboards specific to your mobile app and system. (Examples)

1. Use data from HTTP request response time to speed back-end services

2. NRQL alerts: Filter errors and failures to eliminate noise for certain end points and/or status codes.

3. NRQL alerts: Focus on the percentage of users impacted by errors and failures to check the overall impact on your user base.

4. **Track performance improvements and degradations across app releases**

   Once you’ve established a solid operational base around your app’s performance, it’s time to share progress. **Version trends** in the Mobile UI offers a fast and easy way to note degradations in performance from one app release to the next.

**HOW TO DO IT**

1. Go to rpm.newrelic.com/mobile and select an app, then select App > Version trends.

2. Select the Launch time period to compare versions by day (default), week, or month, or view all available data.

3. Review “Version trends” details to analyze performance, user adoption, and key technical indicators. For additional
suggestions about how to interpret the data, review the examples.

4. Optional: To opt in or out of “Version trends” email reports, select the “Enable version trends email notifications” link.

5. Create an additional Insights dashboard to reflect the performance trends you would like to track over time.

5 Take your crash reporting to the next level

Now it’s time to tackle some of those harder-to-reproduce crashes by adding additional instrumentation to your application and by leveraging crash event trails and Insights.

Go after high-impact crashes with crash analysis

Crashes are an inevitable part of running mobile applications at scale. New Relic gives you the tools to understand your highest-impact crashes. Use the Crash Analysis UI to see which crashes are occurring most often, the percentage of crash-free users, which files and methods are associated with perhaps 80 percent of your crashes, which crashes a particular user just experienced, and more. Start here to understand which crashes are generating the most issues.

HOW TO DO IT

1. Add custom attributes to categorize the highest-value customers, and then use them to segment crashes according to your most valuable audiences. (More)

2. Add breadcrumbs and custom events to produce a detailed trail of events leading up to every crash occurrence (over the last 90 days). You can then view this crash event trail in the Crash Analysis UI. Mobile agents for iOS and Android 5.14 (or later) also allow you to “enable” the collection of event-level data for HTTP requests. (Note: The default is for this to be “disabled.”) These network events will be included in your crash event trail as well. (More)

3. Use Version Trends, in the Mobile UI, to make sure you’re improving the crash rate over subsequent releases and avoiding regressions. (More)

Use breadcrumbs and custom attributes to better reproduce and debug crashes

HOW TO DO IT

1. By using the recordBreadcrumb API (Android, iOS), you can track mid-session changes in state that can help debug crashes, such as:

- Change of connection type
- Change of orientation
- CPU and memory state at key points in the app code

2. By adding custom attributes to existing events, you can ask even more precise questions of your mobile app. Add dimensions such as the following:

- User ID—to react to specific user concerns and better understand which customers/segments are most impacted
- Store ID—to address problems with out-of-date devices, bad Wi-Fi, and so on
- User segments—to better understand which customers and segments are most impacted (logged in vs. non-logged in, etc.)
- Dollars in the cart at session end
- Origin/attribution of entry into the app

3. Standardize custom attribute names across iOS and Android, and align with your website (if monitored by New Relic Browser), so that you can compare and contrast performance across end-user channels.

4. As mentioned earlier, New Relic tracks the last interaction of crashed and non-crashed sessions to provide visibility into what the user viewed before exiting your app (either intentionally or via a crash).

This additional information will help you reproduce crashes and understand app performance in relation to key business metrics.
Bring your key performance indicators to life

Create custom interactions and events to hone in on the most important signals, including user activity for the business-critical aspects of your app. Using these powerful APIs, you can add custom instrumentation to just about anything. The following are some example use cases.

Use custom events to track feature use

- **Track feature use as custom events.** Use the recordCustomEvent API (iOS, Android). Then compare the use of different features using:

  ```sql
  SELECT count(*) FROM MyNewMobileEventType FACET name.
  ```

Use custom events to measure screen-load time

- Create an app-launch timer to measure the time that elapses between a user-tap to launch your app and the point in your app code where the first screen is available. Record the timer value in New Relic using something like the following:

  ```java
  recordCustomEvent(<myMobileTimers>, <appLaunch>, {attributes})
  ```

- Define start/stop timers to measure spinner durations and then record this time in New Relic using something like the following:

  ```java
  recordCustomEvent(<myMobileTimers>, <spinnerOnScreenA>, {attributes})
  ```

- Track specific user actions or funnel steps in the app (like "add to cart") and include the dollar amount as an attribute on that event.

- Measure flows through the application by viewing the related funnel steps via custom events in Insights. Create a timer to track the start and end times of a subflow or an entire flow across multiple funnel steps to understand how long it took users to get through the process.

Use custom interactions to see traces associated with those same screens

- Use the agent API to disable default interactions and focus on custom interactions so that you can review the screen loads that are most crucial to your primary funnel.

- Define the start and stop times of these custom interactions to trace activities associated with individual steps of that funnel (for example, a particular user action or a screen load. (startInteraction [Android], Create and complete interactions [iOS], NRFeatureFlag_DefaultInteractions [iOS], with-DefaultInteractions [Android])

- Track the last interactions of crashed and non-crashed sessions so that you can see what the user viewed before exiting your app (either intentionally or via a crash).

  - **Example query for non-crashed sessions.**

    ```sql
    SELECT count(*) FROM Mobile WHERE sessionCrashed is null FACET lastInteraction SINCE 90 days ago
    ```

  - **Example query for crashed sessions.**

    ```sql
    SELECT count(*) FROM Mobile WHERE sessionCrashed is true FACET lastInteraction SINCE 90 days ago
    ```

Want more user tips?

- View training videos at New Relic University.
- Read the documentation.
- Visit the New Relic Community Forum.
- Check out our Tutorials page.