DataLens Dashboard Builder Guide

Document ID: PRD-DL-020 **Last Updated:** 2024-03-01 **Owner:** DataLens Product Team **Classification:** Public

---

Overview

DataLens Dashboard Builder enables you to create interactive, real-time dashboards for visualizing metrics, logs, and traces from your infrastructure and applications. This guide covers dashboard creation, widget configuration, and sharing options.

Getting Started

Creating a Dashboard

1. Navigate to **DataLens → Dashboards → New Dashboard**
2. Enter dashboard name and description
3. Select a folder for organization
4. Click **Create**

Dashboard Settings

```
dashboard:
  name: Production Overview
  description: Real-time production metrics
  folder: /team/platform
  refresh_interval: 30s
  time_range:
    default: last_1_hour
    options:
      - last_15_min
      - last_1_hour
      - last_24_hours
      - last_7_days
  variables:
    - name: environment
      type: dropdown
      values: [production, staging, development]
      default: production
```

Widget Types

Time Series Graph

Display metrics over time with multiple visualization options.

```yaml
widget:
  type: time_series
  title: Request Rate
  query:
    metric: http_requests_total
    aggregation: rate
    interval: 1m
    group_by: [status_code]
  visualization:
    type: line  # line, area, bar
    legend: bottom
    y_axis:
      label: Requests/sec
      min: 0
    colors:
      "200": green
      "4xx": yellow
      "5xx": red
```

Stat Panel

Single value display for key metrics.

```yaml
widget:
  type: stat
  title: Active Users
  query:
    metric: active_users_count
    aggregation: last
  thresholds:
    - value: 0
      color: red
    - value: 100
      color: yellow
    - value: 500
      color: green
```

```
  format: number
  prefix: ""
  suffix: " users"
```

Gauge

Circular gauge for percentage or bounded values.

```
widget:
  type: gauge
  title: CPU Usage
  query:
    metric: cpu_usage_percent
    aggregation: avg
  min: 0
  max: 100
  thresholds:
    - value: 0
      color: green
    - value: 70
      color: yellow
    - value: 90
      color: red
  format: percent
```

Table

Tabular data display with sorting and filtering.

```
widget:
  type: table
  title: Top Endpoints
  query:
    metric: http_request_duration_seconds
    aggregation: avg
    group_by: [endpoint, method]
    order_by: avg desc
    limit: 20
  columns:
    - field: endpoint
      title: Endpoint
      width: 300
    - field: method
      title: Method
```

```
      width: 80
    - field: avg
      title: Avg Latency
      format: duration
    - field: count
      title: Requests
      format: number
```

Heatmap

Visualize distributions over time.

```
widget:
  type: heatmap
  title: Request Latency Distribution
  query:
    metric: http_request_duration_bucket
    aggregation: rate
  y_axis:
    scale: logarithmic
    unit: seconds
  color_scheme: YlOrRd
```

Log Panel

Display and search log entries.

```
widget:
  type: logs
  title: Application Logs
  query:
    source: application-logs
    filter: level:error OR level:warn
    fields: [timestamp, level, message, service]
  display:
    wrap_lines: true
    show_time: true
    sort: desc
```

Trace Panel

Visualize distributed traces.

```
widget:
  type: traces
  title: Recent Traces
  query:
    service: api-gateway
    operation: /api/v1/*
    min_duration: 500ms
  display:
    show_service_map: true
    limit: 50
```

Alert List

Display active alerts.

```
widget:
  type: alert_list
  title: Active Alerts
  filter:
    severity: [critical, high]
    state: firing
  display:
    show_description: true
    group_by: service
```

Layout and Positioning

Grid System

Dashboards use a 24-column grid system.

```
layout:
  widgets:
    - id: request-rate
      position:
        x: 0
        y: 0
        width: 12
        height: 8
    - id: error-rate
      position:
```

```
        x: 12
        y: 0
        width: 12
        height: 8
    - id: latency-p99
      position:
        x: 0
        y: 8
        width: 24
        height: 10
```

Responsive Layout

```
layout:
  responsive:
    breakpoints:
      desktop:
        columns: 24
      tablet:
        columns: 12
      mobile:
        columns: 6
    widget_min_width: 6
    widget_min_height: 4
```

Rows

Group widgets into collapsible rows.

```
layout:
  rows:
    - title: Overview
      collapsed: false
      widgets: [request-rate, error-rate, latency]
    - title: Database Metrics
      collapsed: true
      widgets: [db-connections, db-queries, db-latency]
```

Variables and Templates

Dashboard Variables

Create dynamic, reusable dashboards.

```yaml
variables:
  - name: environment
    label: Environment
    type: query
    query:
      metric: up
      label: environment
    include_all: true
    default: production

  - name: service
    label: Service
    type: query
    query:
      metric: up{environment="$environment"}
      label: service
    include_all: true
    multi: true

  - name: interval
    label: Interval
    type: interval
    values: [1m, 5m, 15m, 1h]
    default: 5m
```

Using Variables in Queries

```yaml
widget:
  query:
    metric: http_requests_total{environment="$environment", service=~"$service"}
    aggregation: rate
    interval: $interval
```

Chained Variables

```yaml
variables:
  - name: region
    type: static
```

```
    values: [us-west, us-east, eu-west]

  - name: cluster
    type: query
    query:
      metric: cluster_info{region="$region"}
      label: cluster
    refresh: on_variable_change  # Refresh when region changes
```

Annotations

Event Annotations

Mark important events on time series graphs.

```
annotations:
  - name: Deployments
    datasource: deployment-events
    query:
      event_type: deployment
      environment: $environment
    color: blue
    icon: rocket

  - name: Incidents
    datasource: pagerduty
    query:
      severity: [high, critical]
    color: red
    icon: alert
```

Manual Annotations

```
annotations:
  - name: Maintenance Windows
    type: manual
    events:
      - start: 2024-07-15T02:00:00Z
        end: 2024-07-15T04:00:00Z
        title: Database maintenance
        description: Planned PostgreSQL upgrade
```

Linking and Drill-Down

Data Links

Add links to widgets for drill-down.

```yaml
widget:
  data_links:
    - title: View in Logs
      url: /datalens/logs?service=${__field.labels.service}&time=${__value.time}
      open_in_new_tab: true

    - title: View Traces
      url: /datalens/traces?service=${__field.labels.service}
```

Dashboard Links

Link between dashboards.

```yaml
dashboard:
  links:
    - title: Service Details
      type: dashboard
      dashboard: service-details
      variables:
        service: $service
        environment: $environment
      icon: external-link

    - title: Related Dashboards
      type: folder
      folder: /team/platform
```

Sharing and Permissions

Share Options

```yaml
sharing:
  public:
    enabled: false
```

```yaml
  internal:
    enabled: true
    allow_edit: false
    allow_copy: true

  embed:
    enabled: true
    allowed_domains:
      - *.novatech.com
```

Permissions

```yaml
permissions:
  owner: team-platform

  access:
    - team: engineering
      level: view
    - team: sre
      level: edit
    - user: admin@novatech.com
      level: admin
```

Snapshots

Create point-in-time snapshots for sharing.

```
# Create snapshot via CLI
datalens dashboard snapshot create \
  --dashboard production-overview \
  --time-range "2024-07-15T00:00:00Z/2024-07-15T23:59:59Z" \
  --expire 7d
```

Dashboard as Code

Export Dashboard

```
# Export to YAML
datalens dashboard export \
  --name production-overview \
  --format yaml \
```

```
  --output dashboard.yaml

# Export to JSON
datalens dashboard export \
  --name production-overview \
  --format json \
  --output dashboard.json
```

Import Dashboard

```
# Import from file
datalens dashboard import \
  --file dashboard.yaml \
  --folder /team/platform

# Import with variable substitution
datalens dashboard import \
  --file dashboard.yaml \
  --set environment=production \
  --set team=platform
```

Version Control

```
# .datalens/dashboards/production-overview.yaml
apiVersion: datalens.novatech.com/v1
kind: Dashboard
metadata:
  name: production-overview
  namespace: platform
  labels:
    team: platform
    environment: production
spec:
  # Dashboard configuration
```

Best Practices

Dashboard Organization

1. **Use folders** to organize by team/domain
2. **Consistent naming** convention

3. **Include descriptions** for discoverability
4. **Tag dashboards** for searchability

Performance

1. **Limit widgets** to 15-20 per dashboard
2. **Use appropriate refresh intervals** (not too frequent)
3. **Optimize queries** with proper aggregations
4. **Use caching** for expensive queries

Design

1. **Most important metrics** at top-left
2. **Use consistent colors** across dashboards
3. **Group related widgets** in rows
4. **Provide context** with annotations

Troubleshooting

Common Issues

Issue	Solution
Slow dashboard load	Reduce widget count, optimize queries
No data displayed	Check time range, verify query syntax
Widget showing "N/A"	Verify metric exists, check permissions
Variables not working	Check variable query, refresh variables

Query Debugging

```
widget:
  debug:
    enabled: true
    show_query: true
    show_raw_response: true
```

Keyboard Shortcuts

Shortcut	Action
`d`	Toggle edit mode
`e`	Edit selected widget
`v`	View mode
`r`	Refresh dashboard
`t`	Toggle time picker
`s`	Save dashboard
`?`	Show all shortcuts

Related Documents: Query Language (PRD-DL-025), Alerting Guide (PRD-DL-030), Data Sources (PRD-DL-015)